DOTTORATO DI RICERCA
in
SCIENZE COMPUTAZIONALI E INFORMATICHE
Ciclo XXIII

Consorzio tra Università di Catania, Università di Napoli Federico II,
Seconda Università di Napoli, Università di Palermo, Università di Salerno

SEDE AMMINISTRATIVA: UNIVERSITÀ DI NAPOLI FEDERICO II

ALESSANDRO BIANCO

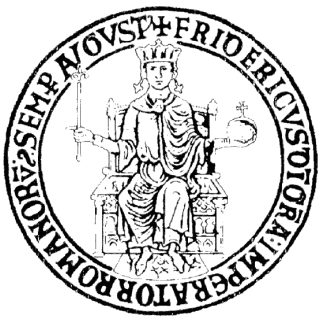
MODELS AND ALGORITHMS FOR FAIRNESS AND PRIORITY IN SCHEDULING

TESI DI DOTTORATO DI RICERCA

`

IL COORDINATORE
Prof. Luigi M. Ricciardi

MODELS AND ALGORITHMS FOR FAIRNESS AND PRIORITY IN SCHEDULING



Alessandro Bianco

Universitá degli Studi di Napoli "Federico II"

Dipartimento di Matematica e Applicazioni "Renato Caccioppoli"

A thesis submitted in fulfilment of the degree of

Doctor in Computer Science

Napoli, November 30, 2010

Supervisor: Prof. Ph.D. Aniello Murano

Advisor: Prof. Ph.D. Marco Faella

Abstract

In this thesis we address the problem of fair and priority scheduling. We focus our attention on systems characterized by an infinite computation, a discrete decomposition of tasks into atomic operation and a known-a-priori set of precedence-constraints among the activities to be performed. We analyze two fairness and two priority specifications related to the frequency of occurrence of activities. We solve the scheduling problem both when the scheduler has complete control upon the system's execution and when the environment can influence the outcome of a scheduling plan.

Acknowledgments

The results presented in this thesis are fruit of the joint work of all the authors of [BFMM09], [BFMM10a], [BFMM10b], which already contain most of the results discussed. We thank Gimbert and Zielonka for pointing out the counterexample of lemma 5.5.2.

Contents

Introduction

Scheduling is the problem of allocating resources to fulfill efficiently a set of partially ordered activities. Depending on the nature of the activities, resources, and on efficient fulfillment one aims at, different scheduling problems arise. Usually, activities are described through a precedence graph, which shows which operation needs to be completed before another one is allowed. Activities may be endowed with properties like the time or cost needed to be completed. Moreover, resources are entities able to work on one activity at a time. Due to their generality, scheduling problems have different applications and the measure of efficiency of a scheduling plan varies. In operations research, activities are jobs that need to be distributed among workers and machines with the purpose to complete the production of goods or the delivery of a service. So, we are interested in minimizing the time or the cost needed to complete all the jobs ([Pin08]) or to distribute evenly the workload on all the available resources ([SL85]). In the study of operative systems, the activities are processes' operations that need to be executed by one or more processors. So, we may want to complete a given program, as soon as possible, hence, we aim at minimizing the execution time. We may also want to ensure that the system is able to serve many little processes as soon as possible, hence, we aim at maximizing the throughput (i.e. the number of processes completed in a time unit), or minimizing the processes' waiting time ([CD73], [SG98]). In computer networks, activities are deliveries of datagrams or responses to service requests that are performed by servers. Here, we may want to evenly distribute the workload on all servers ([LCST98]) or serve the requests fairly or according to some priority policy due to quality of service agreements ([Tan03]). In the study of formal verification, we are interested in ensuring that system's processes, involved in a possible infinite computation, satisfy a given specification. However, it is useful to take into account some reasonable assumptions, like a scheduling policy of the activities, in order to define less complex verification techniques([MP91]). For example, in a distributed mono-processor environment, it is reasonable to hypothesize that each process eventually makes progress or that an activity available for execution is eventually fulfilled ([MP91]). In all the mentioned applications, there are scheduling problems whose aims are to not disregard particular components of the system with respect to the others of the same type (machine in a load-balancing problem, requests in a network problem). When the interest lays on treating equally all the possible activities in an infinite computation, the property sought by the scheduling problem is called fairness. In general a fairness property asks that every process or every process' action is performed sufficiently often whenever it meets a certain requirement along the computation ([Fra86], [AFK87], [VVK05]). Depending on the requirement and on how much " sufficiently " is, different notions of fairness are possible. As long as the scheduling problem is an abstraction that models the fact that an available process cannot be ignored forever, it is not necessary to quantify how often an action is taken. Indeed, simply asking that an action is eventually performed, simplify and makes possible the verification of properties of a system ([Kup95]). Qualitative fairness is subject to different formalizations depending on the condition an action must fulfill in order that it will be eventually taken ([MP91], [Kwi89], [LPS81], [dA99]). However, when the scheduling problem needs to meet the requirements of a real application, quantification becomes important. Indeed, a finite but enormous delay of an available action is undesirable, yet allowed by an qualitative fairness property.

Moreover, adding quantification may also be a reasonable and helpful hypothesis in the verification of system properties ([AH98]). Usually, in the context of formal verification, quantified fairness notions put a bound on the numbers of steps that may elapse before an available action is performed or just ask for the existence of such a bound ([Kwi89], [AH98]). However, this is still not enough for implementable applications, because such a fairness does not guarantee that an equal share of the computation time is reserved to each process or action. [1] In the study of computer networks and operative systems, the suitability of a scheduling policy in serving all the requests equally, often is analyzed through statistical means in dense time models (theory of queues, [Phi03]). These tools allow to analyze the frequency of service of non-precedence-constrained activities when it is not know a priori their time of occurrence, but it is still possible to model their arrival through stochastic variables. The exact computation of the average frequency of service of a group of activities does not only allow to evaluate fairness, but it is also useful to assign a greater priority to a given group by letting it to have an higher frequency of service. However, when the scheduling problem deals with an infinite execution of a given set of precedence-constrained activities, then a statistical analysis is no longer useful. Indeed, it is possible to use a discrete time model, and obtain more precise results than an estimate through statistical mean values.

In this thesis, we introduce a novel study of quantitative, implementation-oriented, fairness scheduling for a set of precedence-constrained activities in discrete time models ([BFMM09]). We define and solve a single-processor scheduling problem for two fairness notions: *(i)* the bounded difference property asks that at every point in the computation the difference between the number of times, two group of activities are performed, is bounded by some finite constant; *(ii)* the balance property asks that the asymptotic frequency of every group of activities is the same. Since the latter property requires the computation of an asymptotic frequency for each group of activities, these values can be considered as priorities associated to the group. Hence, we also solve the related priority scheduling problem ([BFMM10a]). All these scheduling problems are solved in polynomial time through linear programming techniques.

Like in the theory of queues, a scheduling problem may not be characterized by the complete knowledge of the time of availability of each activity. This happens when sources of uncontrollable non-determinism, called environment, act upon the system in such a way to modify the available actions at any given time. In this context, we have a game played between the scheduler and the environment ([KVW01]). The two players alternate moves and have complementary goals, in particular the scheduler's (or system's) aim is to force an infinite computation satisfying the required scheduling property no matter the choices of the opposite player. The system-environment game models have been studied as a mean to prove the correctness of an interactive system's temporal logic specification ([KVW01], [AHK02]). In particular, modal logics, such as LTL, and CTL^*, are useful to express qualitative fairness requirements ([Kup95]). Other logics such as $RCTL$ and $TCTL$ are able to put time bounds between the occurrences of actions ([EMSS90], [ACD90]), however, the most famous modal logics are not able to evaluate exactly the frequency of execution of activities. The models of the theory of queues can also be viewed as real time

[1] For example, if in a set of two actions we guarantee that each one of them is performed at least once every three time steps, then one may performed twice as much as the other. On the other side we may not be able to guarantee that each of them is performed at least once every two times units, due to precedence constraints. However, we may guarantee that they are taken twice every four time units, but this property is not expressible through the previous type of fairness.

stochastic games where the decision of the environment is determined a priori by a stochastic variable representing the rate of occurrence of certain requests. In this thesis we also address and study the problem of quantitative, implementation-oriented, fairness scheduling for a set of precedence-constrained activities in discrete time models, when the environment can influence the choices of the scheduler by acting at predetermined instants in time ([BFMM10a]). We prove that determining whether the scheduler can force one of the discussed scheduling properties is a $CO - NP$ complete problem. In the proof of the latter result, we make use of a property of the game model called half-positionality ([Kop06]). It states that the environment can play optimally without keeping into account the history of the computation developed so far. Since the same property may be used to prove similar results for other scheduling problems on games, we investigated what conditions are required or are sufficient for its validity. We determined a new sufficient condition ([BFMM10b]) that allows to classify as half-positional a wider class of game goals.

Applications

The thesis focuses on quantitative, fairness and priority scheduling problems for systems where the set of activities to be repeated ad infinitum [2] and their precedence relation is known a priori. The activities are considered atomic and equally expensive (same length or cost). However, since they are organized in groups and the scheduler tries to treat all groups equally, one may model activities of multiple duration with a sequence of activities belonging to the same group. Unfortunately, due to the necessity to find some atomic activity component, all the preemptive applications, where the activity can be halted at any point in a dense time, are ruled out by our model.

In software applications, our scheduling solution may be used when all processes or tasks involved in an infinite computation are known a priori, in this case every process or task can be decomposed in a sequence of activities belonging to the same group. For example, a security control systems has to continuously perform a series of predetermined measures, observations and computations. So, one may want to ensure that all control tasks are performed equally often or with a given priority. However, if the control tasks require a small time, a scheduling plan may be useless, since it may be very easy to perform all controls adequately often without a plan. If the control tasks require a lengthy time for a task and also a lengthy time for moving from a task to another, then a plan ensuring that all tasks are performed equally often may be valuable. A concrete application is a concurrent program accessing to peripherals that can be used in one-hour slots, such as a telescope. In operations research applications, we can solve a variant of the traveling salesman problem, where the aim of an infinite traveling schedule is visiting equally often all places. In this case, the activities are the visit of a given site and the precedence relation is their spatial order. For example, the traveling salesman may want to periodically visit cities to look for new contacts or clients, or a controller may want to periodically monitor some characteristics of the given areas. As further examples, we may solve a scheduling problem for crop rotation in agriculture, where the activities are the culture of a certain plant, the relations are the optimal order of plant culture and the fairness scheduling aims at planting all the cultures equally often in the long run. Also, we may imagine planning a diet with the aim to receive a balanced intake

[2] repeated for a very long undetermined time

of all nutrients, in this case taking a particular nutrient may be considered as an activity and the constraint on the type of food one may eat at different time of day may provide the order relation among them.

In all mentioned applications, there may be sources of periodic non-determinism: in a control system peripherals may be periodically rebooted or be subject to maintenance and not be available for a certain time, the salesman may try to accommodate the requests of clients, the weather may be too cold or too hot for a certain crop, one may be invited to dinner every Sunday, thus disabling some food choices for that day. In all these situations, the game model may be used to determine the existence of a fairness or priority scheduling plan.

Preliminaries

Graphs are mathematical models useful to visually represent and formally analyze a wide class of systems. A transition table of an automaton, the control flow of an algorithm, the connections of gates in a logic circuit, the geographical position of areas, the order relation of activities in a plan, the structure of a complex software architecture are just examples of applications of graph models.

Formally, a graph is a structure composed by a set V of entities, called nodes, and a relation $E \subseteq V \times V$ whose elements are called edges. So, for example, in a geographical representation the set of nodes is the set of cities and the edges represent the allowed paths between them. The nodes and the edges may be endowed with properties expressible through functions from the set of nodes and edges respectively to the set of properties. For example, each city has a given number of inhabitants and each path has a length. However, when we attempt to model a property of the edges we may find out there are multiple ways to relate the same two nodes and these ways differ from the value of the property. In such a situation we need to relate not only the nodes but the property as well. A weighted graph is a structure composed by a set V of nodes and a relation $E \subseteq V \times P_E \times V$ where P_E is a set of relevant properties. In the example of the geographical representation P_E may be the set of all distances of paths expressed in meters.

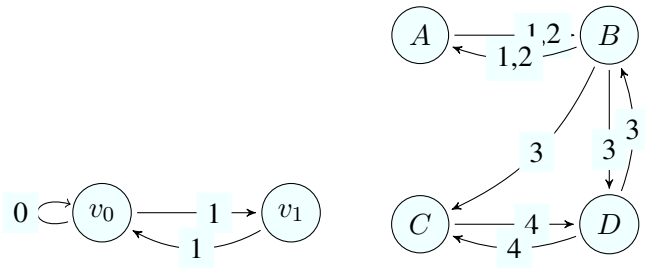
In this thesis, we make use of weighted graphs to represent the order relation of activities for a scheduling problem. The nodes represent the states of the plan and determine what activities are allowed next, the edges represent atomic activities of equal cost: they are allowed at a given state and, once performed, bring the system to a new state. The activities are organized in a finite number of groups, representing for example the process they belong to or the task they contribute to fulfill. These groups are represented by integer number in a set $[k] = \{0, \ldots, k\}$ and constitute the property associated to the edges. We call such a graph, k-colored graph, since we imagine to visualize each edge with the color of the group they belong to.

Definition 0.0.1. k-colored graph

A k-colored graph is a structure $G = (V, E)$ such that V is a finite set of nodes and $E \subseteq V \times [k] \times V$ is a set of colored edges.

Example 0.0.1. *Suppose there are two infinite processes 0 and 1 in competition for the use of a the only processor and for the use of a shared variable x. Suppose process 0 continuously acquires x, performs one atomic operation and then releases it, and that process 1 continuously acquires x, performs two atomic operations and then releases it. If we disregard the operation of acquiring and releasing the variable, we can determine two system states.* (i) *A state v_0 where variable x is free and both processes can lock it and perform their operations* (ii) *a state v_1 where variable x is locked by process 1 and, hence, only the latter process can perform its second operation. The k-colored graph representing the order relation of the activities is visualized in figure 1(a) Observe that the edge colors represent the process the activity belongs to.*

Example 0.0.2. *Consider a robot patrolling an geographical area and moving between four points A, B, C, D. The activities the robot has to perform are taking picture of four sorrounding areas: 1 the sea, 2 the cliffside, 3 the hill 4 the forest. The road from A to B is on the coast and the robot*



(a) (b)

Figure 1: Example graphs.

can see the sea and the cliffside. The road from B to C is on the hill, due to its slope the robot can only traverse it from B to C, on this road the robot can see the hill and the sea. The road from C to D is in the forest, hence the robot can only see the forest. The road from D to B is on the other side of the cliffside, hence, the robot can only see the hill. (See Figure 1(b))

Systems progress in time by performing activities and changing states, such progress is called run or execution of the system and, on a graph, it is represented by an infinite path. Some paths may have a desirable property, while others do not meet the system's specifications. A rule describing what paths are desirable is called system's goal: it may be a temporal logic formula or explicitly the set of allowed infinite paths among those contained in the graph. In this thesis, we are interested in expressing properties related to the frequency of execution of groups of activities, or expressed in different term, the frequency of occurrence of colors along an infinite path. Hence, we express a goal as a set of infinite sequences of colors.

Definition 0.0.2. Winning Condition

A goal or winning condition W on a k-colored arena, is a set of infinite sequences of colors in $[k]$, i.e., $W \subseteq [k]^\omega$.

Example 0.0.3. *In Example 0.0.1, we may want that in four time units, each process uses the processor for 2 units. Then the system's goal is $W = \{0011, 0110, 1100, 0101, 1010, 1001\}^\omega$. Observe that the scheduler can force a path in $\{0011, 0110, 1100\}^\omega$ thus meeting the goal.*

When the decision of the particular activity to be performed at every node cannot be always determined by the system, it is useful to introduce a function that maps each node to the entity, the choice is up to. In this way, the graph becomes an arena where multiple decision entities, called players, influence the outcome of the plan. When the purpose of the model is the verification of a system property, it is sufficient to group all factors, but the system, as one unique source of nondeterminism called environment. In this thesis, we use a 2-players weighted arena with the purpose to determine whether a scheduler can force a plan with a given property no matter how much adverse the environment is.

Definition 0.0.3. k-colored arena

A k-colored arena is a structure $A = (V_0, V_1, v_{ini}, E)$ where V_0 and V_1 are two disjunct set of nodes, v_{ini} is an initial node and $E \subseteq (V_0 \cup V_1) \times [k] \times (V_0 \cup V_1)$ is the set of edges.

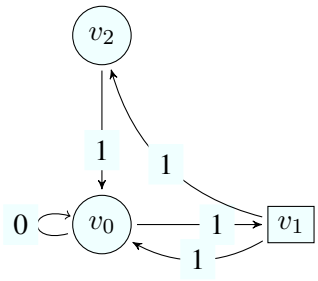


Figure 2: Example arenas.

The node in the set V_i are said belonging to player i. We graphically represents an arena in the same way as a graph, with the exception that nodes in V_0 are represented as circles and nodes in V_1 are represented as squares.

Example 0.0.4. *Suppose that processor* 1 *of Example 0.0.1, after performing the first operation can non-deterministically choose to perform 1 or 2 more operations before releasing the shared variable. Hence, in state* v_1*, the system has no control, but the environment has the ability to choose whether to perform another operation and release the variable or perform the same operation and bring the system in a third state* v_2 *where the variable is still owned by processor* 1*. See Figure 2.*

Also on arenas, a system's run is represented by an infinite path on the graph. The only difference is that at every point along the run the choice of the next edge does not depend necessarily on the system.

When an arena is paired with a winning condition, then we have a game between the system and the environment. They alternate moves along the game arena, and construct a run. If the run satisfies the winning condition the system wins, otherwise the environment, considered as malicious, wins.

Definition 0.0.4. k-colored game

A k*-colored game is a structure* $G = (A, W)$ *composed by a* k*-colored arena* A *and a* k*-colored winning condition* W*.*

We say that a system can force a winning condition W, when the scheduler has a criteria to make choices at every step of the execution, in such a way that the resulting infinite run satisfies W. A criteria that describes the choice made by one player at every point in the execution is a function called strategy. It associates to every finite path ending with a node, that belongs to that player, the next activity to be performed. So it is interesting to determine whether a player has a strategy that allows him to pursue its goal.

0.1 Basic Notation

Here, we introduce some basic preliminary notation that is used through the thesis.

Let X be a set then we use the following notations:

1. A word on X is a finite or infinite sequence of elements of X.
2. The word with no symbols, called also the empty word, is denoted with ε.
3. A language on X is a set of words on X.
4. For an element $x \in X$ we use x also to mean the language $\{x\}$ when the meaning is clear from the contest.
5. X^i is the set of all sequences of elements of X with length i, i.e., the cartesian product of X with itself i times. We set $X^0 = \{\varepsilon\}$ where ε is the empty word
6. X^* (resp., x^ω, $x^{+\infty}$) is the set of all finite (resp., infinite, finite or infinite) words on X.
7. $X^{n\times m}$ is the set of all matrices of dimension $n \times m$ with elements belonging to X.
8. $|X|$ is the cardinality of X, i.e., the number of its elements.
9. 2^X is the set of all subsets of X.

Automaton recognizes sets of words and are useful to determine the properties of such sets.

1. A finite state automaton is a tuple (X, Q, δ, q_0, F) where X is an alphabet, Q a set of states, $q_0 \in Q$ an initial state, $F \subseteq Q$ a set of final states and $\delta : Q \times X \to 2^Q$ a transition function.
2. A run of the automaton on a sequence $x_1 \dots x_k \in X^*$ is a sequence $q_0, \dots, q_k \in Q^*$ such that or each $i \in \{1, \dots, k\}$ we have $q_i \in \delta(q_{i-1}, x_i)$.
3. A word $x \in X^*$ is said accepted by the automaton if there exists a run $q_0, \dots, q_k \in Q^*$ on x ending in a final state $q_k \in F$.
4. A language $L \subseteq X^*$ is said regular if there exists a finite state automaton that accepts all and only the words belonging to it.

The set of numbers are described through the following notations

1. $\mathbb{N}$ is the set of natural number with number 0 included, $\mathbb{N}_{-0} = \mathbb{N} \setminus \{0\}$
2. $\mathbb{Z}$ (resp., $\mathbb{Q}$, $\mathbb{R}$) is the set of integer (resp., rational, real) numbers.
3. For a natural $k \in \mathbb{N}_{-0}$, we use the notation $[k] = \{1, \dots, k\}$ to denote the set of the first k non-zero natural numbers.
4. Given two real numbers $a, b \in \mathbb{R}$, we use the notation $[a, b] = \{r \in \mathbb{R} \mid a \leq r \leq b\}$ to denote the set of real numbers between a and b.

Let $A \in \mathbb{R}^{n\times m}$, $B \in \mathbb{R}^{n\times 1}$ be two matrices and $x \in X^{m\times 1}$ be a variable vector, then we use the following notation.

1. $a_{i,j}$ is the elements of A on row i and column j.

2. A is called a square matrix if the dimension n and m are equal.

3. The determinant of a square matrix $A \in \mathbb{R}^{n \times n}$ is a real number $det(A)$ given inductively by the Liebniz formula *(i)* for $n = 1$, $det(A) = a_{1,1}$ *(ii)* for $n > 1$ $det(A) = (-1)^{i+1} a_{1,i} \cdot det(A_{1,i})$ where $A_{i,j}$ is the matrix obtained from A by removing the i-th row and the j-th column.

4. A is singular if it has a null determinant.

5. A^T is the transpose of matrix A, i.e., that matrix $A^T \in X^{m \times n}$, such that its element of position i, j is $a'_{i,j} = a_{j,i}$.

6. $\mathbf{0}$ is the matrix with all null components, $\mathbf{1}$ is the matrix with all elements equal to 1, when the dimension is clear from the context.

7. $Ax = B$ is a linear system, given by the set on equalities $\sum_{j=1}^{m} a_{i,j} x_j = b_i$. A is called coefficient matrix, and B is called matrix of known values.

8. A linear system $Ax = B$ is said homogeneous if $B = \mathbf{0}$.

9. A matrix $C \in \mathbb{R}^{m \times 1}$ is a solution of the system $Ax = B$ if $A \cdot C = B$.

10. For $\sim \in \{\leq, <, \geq, >\}$, $Ax \sim B$ is a system of linear inequalities given by the set of the inequalities $\sum_{j=1}^{m} a_{i,j} x_j \sim b_i$

We now recall some results on linear systems and systems of linear inequalities.

1. Cramer's Theorem. Given a linear system $Ax = B$, if A is a square non-singular matrix then a the system admits solutions, one such solution z has components $z_i = \frac{det(A_i)}{det(A)}$ where A_i is the matrix obtained from A by replacing the i-th column with the column matrix B.

2. If a system of linear inequalities contains only integer coefficients and has a solution, then it has also a rational solution. [NW88]

3. Given a system of linear inequalities with real variables and with no $>, <$ inequality sign, then we can compute a rational solution of the system in polynomial time in the size of the system and in the logarithm of the maximum value of the coefficients, or at least determine that such a solution does not exists. [NW88]

4. Given a system of linear inequalities with integer variables, determining a solution of the system is an NP-complete problem [NW88].

Part I

Problems on Graphs

1

Preliminaries on Graphs

Contents

In this part of the thesis, we deal with colored graph, representing the order relation of activities, and with colored goals, representing a specific scheduling requirement. Under the hypothesis that the scheduler has complete decision power, we define polynomial algorithms for the construction of a plan for some specific fairness and priority conditions. This part is divided in three chapters, the second dealing with fairness scheduling and the third with priority scheduling. The colored goals are defined in the respective chapters. IN this chapter, we introduce preliminary notations on colored graphs that will be used through all the thesis.

1.1 Colored Graphs

Like defined in the preliminaries of the thesis, a k-colored graph is a structure composed by a finite set of nodes V, a set of edges $E \subseteq V \times [k] \times V$. When we highlight a starting state, the graph is said initialized.

Definition 1.1.1. k-colored Graphs

1. *A k-colored graph is a structure $G = (V, E)$ such that V is a finite set of nodes and $E \subseteq V \times [k] \times V$ is a set of colored edges.*
2. *An initialized k-colored graph is a structure $G = (V, E, v_{ini})$, such that (V, E) is a k-colored graph and $v_{ini} \in V$ is a starting node.*
3. *For each edge $(v, a, u) \in E$, v is said starting node, or source of e, u is said ending node or destination of e, a is said color of e.*
4. *For each node $v \in V$, the set of edges starting from v is denoted as $v^{\rightarrow} = \{(v, a, u) \in E\}$, the set of edges ending in v denoted as $v^{\leftarrow} = \{(u, a, v) \in E\}$.*
5. *For each color $a \in [k]$, the set of edges colored by a is denoted as $E(a) = \{(v, a, u) \in V\}$*
6. *A subgraph of G is a graph $G' = (V', E')$ such that $V' \subseteq V$ and $E' \subseteq E$.*

We can imagine to walk through the path from a node to another through an edge relating them. Thus a sequence of edges represent a path on the graph.

Definition 1.1.2. Colored paths

Let $G = (V, E)$ be a k-colored graph.

1. *A finite (resp., infinite) path on G is a finite (resp., infinite) sequence of colored edges $\rho = e_1 \dots e_m \in E^*$ (resp., $\rho = e_1 \dots e_m \dots \in E^\omega$) such that consecutive edges are related through a node, i.e., for all $i \in [1, m-1] \cap \mathbb{N}$, (resp., for all $i \in \mathbb{N}_{-0}$) the ending node of e_i is equal to the starting node of e_{i+1}.*
2. *A finite (resp., infinite) path $\rho = e_1 \dots e_m \in E^*$ (resp., $\rho = e_1 \dots e_m \dots \in E^\omega$) can be represented as a sequence of alternating nodes and colors $\rho = v_1 a_1 \dots v_m a_m v \in (V \times [k])^* \times V$ (resp., $\rho = v_1 a_1 \dots v_m a_m \dots \in (V \times [k])^\omega$) where each v_i is the starting node of e_i and a_i is its color. When the colors are non-relevant in the contest, we also represent a path with the only sequence of nodes $v_1 \dots v_m, v$ (resp., $v_1 \dots v_m \dots$).*

3. *Given a path $\rho = v_1 a_1 \dots v_m a_m \dots \in (V \times [k])^\omega \cup (V \times [k])^* \times V$, the color sequence of ρ is the sequence of colors of its edges $Col(\rho) = a_1 \dots a_m \dots$.*

4. *The starting node of the first edge of a path ρ is said starting node of the path. A path ρ starting at node v is also said v-path.*

5. *The ending node of the last edge of a finite path ρ is said ending node of the path.*

6. *The length of a finite path ρ is the number of its elements and is denoted as follows: $|\rho| = |e_1 \dots e_m| = m$. The length of an infinite path ρ is infinite $|\rho| = \omega$.*

7. *Given a finite path ρ and an edge e, we denote by $|\rho|_e$ the number of occurrences of the edge e in ρ.*

8. *Given a finite path ρ and an color a, we denote by $|\rho|_a$ the number of occurrences of the edges colored by a in ρ.*

9. *A path is said simple when it does not pass twice through the same node, unless at the beginning and at the end i.e., $\rho = v_1 e_1, \dots, v_m e_m v_{m+1}$ is simple if $v_i \neq v_j$ for all $(i, j) \in [m+1]^2 \setminus \{(1, m+1), (m+1, 1)\}$.*

When in a graph it is possible to reach a node from any other node, we have a strong connection property we use in course of the proofs

Definition 1.1.3. Connection properties

1. *A k-colored graph $G = (V, E)$ is said connected if for every pair of nodes $(v, u) \in V^2$ there exists a path connecting u to v (i.e. either a path from v to u or a path from u to v).*

2. *A k-colored graph $G = (V, E)$ is said strongly connected if for every pair of nodes $(v, u) \in V^2$ there exists a path from v to u.*

3. *A connected component of a graph G is any strongly connected subgraph G', such that there does not exist another strongly connected subgraph $G'' \neq G'$ such that G' is also a subgraph of G''.*

4. *Given a graph G, the strongly connected components of G can be computed in time $\theta(E \times V)$ with Tarjan's algorithm ([CLRS01]).*

Paths can be decomposed and composed like any sequence of characters belonging to some alphabet.

Definition 1.1.4. Prefixes and concatenations

1. *The n-th edge of ρ is denoted as $\rho(n)$.*

2. *The prefix of length n of a path $\rho = e_1 \dots e_m \dots$ (resp., color sequence $x = c_1 \dots x_m \dots$) of length $m > n$ is the path given by the first n edges and is demoted by $\rho^{\leq n} = e_1 \dots e_n$ (resp., $x^{\leq n} = c_1 \dots c_n \dots$).*

3. *The suffix after n steps of a path $\rho = e_1 \dots e_m \dots$ (resp., color sequence $x = c_1 \dots x_m \dots$) of length $m > n$ is the path given by all but the first n edges and is denoted by $\rho^{>n} = e_{n+1} \dots e_m \dots$ (resp., $x^{\leq n} = c_1 \dots cm \dots$).*

4. *A path $\rho = e_1 \dots e_n \dots$ is said consecutive to a finite path $\rho' = e'_1 \dots e'_m$ if the ending node of ρ' is equal to the starting node of ρ.*

5. *If ρ is consecutive to ρ' the concatenation of ρ' with ρ is a new path obtained by adding after ρ' the sequence of edges of ρ. Such concatenation is denoted as $\rho' \cdot \rho = \rho'\rho = e'_1 \dots e'_m e_1 e_2 \dots$.*

6. *Consider a finite (resp., infinite)sequence of paths $\rho_1, \dots \rho_m$ (resp., $\rho_1 \dots \rho_m \dots$), such that for all $i \in [1, m-1] \cap \mathbb{N}$ (resp., for all $i \in \mathbb{N}_{-0}$) ρ_{i+1} is consecutive to ρ_i. Then we denote a finite (resp., infinite) concatenation of such paths as $\prod_{i=1}^{m} \rho_i$ (resp., $\prod_{i=1}^{\omega} \rho_i$).*

Among paths, those that are consecutive with themselves are particularly meaningful. Although finite, they can be repeated ad infinitum and form an infinite path, thus providing a succinct way to represent an infinite computation.

Definition 1.1.5. Loops
Let G be a k-colored graph.

1. *A loop is a finite path whose starting node is equal to its ending node. So, it holds that $\rho = v_1 a_1 \dots v_m a_m v_{m+1} \in (V \times [k])^* \times V$ is a loop if $v_1 = v_{m+1}$.*

2. *A loop $\sigma = v_1 a_1 \dots v_m a_m v_{m+1} \in (V \times [k])^* \times V$ is said simple if it pass only one time through the intermediate nodes and only twice through the extreme nodes, i.e., for all $(i, j) \in [m+1]^2 \setminus \{(1, m+1), (m+1, 1)\}$ it holds $v_i \neq v_j$.*

3. *The concatenation of a loop σ with itself $i \in \mathbb{N} \cup \{\omega\}$ times is denoted as σ^i.*

4. *An infinite path ρ is said periodic if there exists a loop σ such that $\rho = \sigma^\omega$*

5. *An infinite path ρ is said eventually periodic if there exists a finite path ρ' and a loop σ consecutive to ρ' such that $\rho = \rho' \cdot \sigma^\omega$.*

To our purposes it is useful to relate a path with the occurrences of colors on them.

Definition 1.1.6. Color's representation
Let $x \in [k]^*$ be a finite color sequence, and let ρ be a path with color sequence x, then we define the following notations.

1. *The number of occurrences of a color $a \in [k]$ on x (resp., ρ) is denoted with $|x|_a \in \mathbb{N}$ (resp., $|\rho|_a = |x|_a$).*

2. *The vector of occurrences of colors on x (resp., ρ) is denoted as follows: $\overline{N_{um}}(x) = (|x|_1, \dots, |x|_k) \in \mathbb{N}^k$ (resp., $\overline{N_{um}}(\rho) = \overline{N_{um}}(x)$).*

3. *The difference of occurrences of two colors $a, b \in [k]$ on x (resp., ρ) is denoted with $d_{iff\,a,b}(x) = |x|_a - |x|_b \in \mathbb{R}^k$ (resp., $d_{iff\,a,b}(\rho) = d_{iff\,a,b}(x)$).*

4. *The vector of differences between the occurrences of all colors in $[k]$ and the occurrences of a fixed color $a \in [k]$ on x (resp., ρ) is denoted with $\overline{d_{iff}}_a(x) = (d_{iff\,1,a}(x) \dots d_{iff\,k,a}(x)) \in \mathbb{R}^k$ (resp., $\overline{d_{iff}}_a(\rho) = \overline{d_{iff}}_a(x)$).*

5. *We simply call difference vector of x (resp., ρ), the vector $\overline{d_{iff}}(x)$ given by the the differences between the number of occurrences of a given colors and the number of occurrences of the reference color k, i.e., $(d_{iff\,1,k}, \dots, d_{iff\,k-1,k}) \in \mathbb{R}^{k-1}$.*

6. *The difference matrix of x (resp., ρ) is the matrix $\underline{d_{iff}}(x) \in \mathbb{R}^{k \times k}$ (resp., $\underline{d_{iff}}(\rho)$) whose element on row i and column j is $d_{iff\,i,j}(x)$.*

1.2 Composition and Segmentation

Loops may constitute a succinct way to represent infinite paths, when there is some form of periodicity in the sequence of edges. We also have a more succinct way to represent the occurrences of colors along a path by partitioning the color sequence in the color vectors associated to the loops. For example, in a 2-colored graph, if σ is a loop with color vector $(1, 2)$, we can represent the color sequence of σ^ω with $(1, 2)^\omega$. Although we lose some information on the actual succession of colors, this does not necessarily stop us from evaluating a fairness or priority requirement on the path. Indeed, all color sequences, represented by $(1, 2)^\omega$, achieve a priority scheduling that gives to the second color twice as much priority as the first. Hence, investigating the possible loops in a graph may provide insights on the existence of a scheduling plan. Since the number of loops in a graph is infinite, we restrict our observations to their basic components: the simple loops. The purpose of this preliminary section is to show how paths and loops can be decomposed in the simple loops contained within.

A loop is a composition of a finite sequence of simple loops $\mathcal{T}$ f it is obtained by using all and only the edges of $\mathcal{T}$ as many times as they appear in $\mathcal{T}$.

Definition 1.2.1. Composition of Loops

The loop σ is a composition of $(\sigma_1, \dots, \sigma_l)$ if, for all edges e, it holds $|\sigma|_e = \sum_{i=1}^{l} |\sigma_i|_e$.

Given a sequence of loops $\mathcal{T} = (\sigma_1, \dots, \sigma_l)$, we can define a sequence $\mathcal{T}' = (\sigma'_1, \dots, \sigma'_{l'})$ of distinct loops appearing in $\mathcal{T}$ and the number c_i of times σ'_i appears in $\mathcal{T}$. Then, if a loop σ is a composition of $\mathcal{T}$, we also say that σ is a linear composition of $\mathcal{T}'$ with coefficients $c_1, \dots, c_{l'}$, and we have $|\sigma|_e = \sum_{i=1}^{l'} c_i \cdot |\sigma'_i|_e$.

Definition 1.2.2. Natural linear combination

Given a finite set of loops $\mathcal{L} = \{\sigma_1, \dots, \sigma_l\}$, the natural linear combination (n.l.c. for short) of $\mathcal{L}$ with non-null coefficients $c_1, \dots, c_l$ is the set $\mathcal{T} = \{(\sigma_1, c_1), \dots, (\sigma_l, c_l)\}$.

Definition 1.2.3. Composition of Natural Linear Combination

The loop σ is a composition of a natural linear combination $\{(\sigma_1, c_1), \dots, (\sigma_l, c_l)\}$) if, for all edges e, it holds $|\sigma|_e = \sum_{i=1}^{l} c_i \cdot |\sigma_i|_e$.

A generic path ρ can be decomposed in loops, however, due to the fact that the edges do not always coincide, an extra simple path is needed.

Definition 1.2.4. Composition of loops and a path

The path ρ is a composition of a sequence of simple loops $(\sigma_1, \ldots, \sigma_l)$ and a simple path ρ' if, for all edges e, it holds $|\rho|_e = |\rho'|_e + \sum_{i=1}^{l} |\sigma_i|_e$.

We show, now, that a path ρ can always be decomposed in a sequence of simple loops and a simple path, and that when ρ is a loop, the simple path is empty. Our proof is through the construction of an algorithm that computes a particular composition. Since the composition is obtained by scanning the path from starting node to ending node and removing the loops along the way, like segments, it is called *quasi-segmentation*. The definition of the quasi-segmentation is recursive, so it can be immediately translated in a recursive algorithm.

Definition 1.2.5. Quasi-Segmentation

Given a path ρ, we recursively define on the length of ρ, the quasi-segmentation and the rest of the quasi-segmentation as follows.

1. *The quasi-segmentation is a sequence of simple loops, and the rest is a simple path. The rest is either empty or ending with the last node of ρ.*

2. *If ρ has length 1 and it is not a loop, then the quasi-segmentation is the empty sequence and the rest is ρ itself.*

3. *If ρ has length 1 and it is a loop, then the quasi-segmentation is ρ itself and the rest is the empty sequence.*

4. *If ρ has size n, let $\rho' = \rho^{\leq n-1}$, let $\sigma_1, \ldots, \sigma_n$ be the quasi segmentation of ρ' and r be its rest. Consider the path r' obtained by extending r with the last edge of ρ (this can be done because the last node of r is the last node of ρ').*

   (a) *If r' does not contain a loop, then the quasi-segmentation of ρ is $\sigma_1, \ldots, \sigma_n$ and the rest is r'. Observe that r' ends with last node of ρ.*

   (b) *If r' contains a loop σ, this loop is due to the last added edge, i.e., $r' = r''\sigma$. In this case the quasi-segmentation of ρ is $\sigma_1, \ldots, \sigma_n, \sigma$ and the rest is r''. Observe that if r'' is not empty, then it ends with the first node of σ which is also equal to its last node, hence it equal the last node of r', and to the last node of ρ.*

The following lemma shows that the quasi-segmentation and the rest constitutes actually a decomposition of a path. It also shows that a loop is a composition of its quasi-segmentation only, since the rest is always empty.

Lemma 1.2.1. *Quasi-Segmentation Lemma*

A path ρ is a composition of its quasi-segmentation and the rest of its quasi-segmentation. If the rest is not empty, then it starts with the starting node of ρ (and ends with the ending node of ρ).

Proof. The proof is by induction on the length of ρ. The base case is trivial either if ρ is a loop or if ρ is not a loop. For the inductive case: let $n = |\rho|$, $\rho' = \rho^{\leq n-1}$, $\sigma_1, \ldots, \sigma_l$ be the quasi-segmentation of ρ' and r be its rest. By inductive hypothesis, ρ' is a composition of $\sigma_1, \ldots, \sigma_l$ and r, moreover if r is not empty, it starts with the starting node of ρ'. Consider the path r' obtained by extending r with the last edge of ρ, i.e., $r' = r \cdot \rho(n)$.

1. If r' does not contain a loop, then the quasi-segmentation of ρ is $\sigma_1, \ldots, \sigma_n$ and the rest is r'. Observe that r' starts with the starting node of ρ' which is also the starting node of ρ. Moreover, for all edges $e \neq \rho(n)$ we have $|\rho|_e = |\rho'|_e = |r|_e + \sum_{i=1}^{l} |\sigma_i|_e = |r'|_e + \sum_{i=1}^{l} |\sigma_i|_e$. Also $|\rho|_{\rho(n)} = |\rho'|_{\rho(m)} + 1 = |r|_{\rho(n)} + 1 + \sum_{i=1}^{l} |\sigma_i|_{\rho(n)} = |r'|_{\rho(n)} + \sum_{i=1}^{l} |\sigma_i|_{\rho(n)}$. Hence, ρ is a composition of the quasi-segmentation and its rest.

2. If r' contains a loop σ, this loop is due to the last added edge, i.e., $r' = r''\sigma$. In this case the quasi-segmentation of ρ is $\sigma_1, \ldots, \sigma_n, \sigma$ and the rest is r''. Observe that if r'' is not empty, then it starts with the starting node of ρ' which is also the starting node of ρ. Moreover, for all edges $e \neq \rho(n)$ we have $|\rho|_e = |\rho'|_e = |r|_e + \sum_{i=1}^{l} |\sigma_i|_e = |r''|_e + |\sigma|_e + \sum_{i=1}^{l} |\sigma_i|_e$. Also $|\rho|_{\rho(n)} = |\rho'|_{\rho(m)} + 1 = |r|_{\rho(n)} + 1 + \sum_{i=1}^{l} |\sigma_i|_{\rho(n)} = |r'|_{\rho(n)} + \sum_{i=1}^{l} |\sigma_i|_{\rho(n)} = |r''|_{\rho(n)} + |\sigma|_{\rho(n)} + \sum_{i=1}^{l} |\sigma_i|_{\rho(n)}$. Hence, ρ is a composition of the quasi-segmentation and its rest.

□

Since in a loop the extremes coincide, and since the rest of a quasi-segmentation is a simple path, the rest of a quasi-segmentation of a loop is necessarily empty. Hence the following corollary holds

Corollary 1.2.1. *A loop σ is a composition of its quasi-segmentation.*

In the following sections, it is useful to quantify the least number of loops contained in the decomposition of a loop. If a loop of length n contains at most m nodes, then every loop in a composition can have at most length m because it needs to pass only once through every intermediate node. Hence, a composition will contain at least $\lceil \frac{n}{m} \rceil$ simple loops. Since we proved that every loop admits a composition of simple loops, the following corollary holds.

Corollary 1.2.2. *A loop σ of length n containing m distinct nodes is a composition of at least $\lceil \frac{n}{m} \rceil$ simple loops.*

For an infinite path, we can define a quasi-segmentation given by the limit of the quasi-segmentation of all its finite prefixes.

Definition 1.2.6. Infinite quasi-segmentation

The quasi-segmentation of an infinite path ρ is the infinite sequence of loops given by the limit of the quasi-segmentation of $\rho^{\leq n}$, for $n \to +\infty$. The quasi-segmentation of an infinite path has no rest.

So far we proved that given a loop, it is always possible to decompose it, in a sequence of simple loops. However, we do not know whether, given a sequence of simple loops, they can be composed in order to construct a loop containing them all. In general, the answer is negative, since the loops may belong to different connected components of the graph and not be reachable to one another. Hence, a key requirement is the connection between the loops.

Definition 1.2.7. Connected Loops

Two loops σ, σ' in G are connected if there exists a path from a node of σ to a node of σ', and vice-versa. A set or sequence $\mathcal{L}$ of loops in G is connected if all pairs of loops in $\mathcal{L}$ are connected.

However, just having connect loops does not ensure the possibility to construct a loop using just their edges. Indeed, we may need to use the paths connecting them. Since all paths connecting the loops may have at least an edge outside the loops, in general we cannot construct a composition. Hence, we have to ensure that all the paths connecting the loops use only edges of the loop, and that they can pass from a loop to another only through shared nodes.

Definition 1.2.8. Overlapping Loops

Two loops in G are overlapping if they have a node in common. A set or sequence $\mathcal{L}$ of loops in G is overlapping if for all pairs of loops $\sigma, \sigma' \in \mathcal{L}$ there exists a sequence $\sigma_1, \dots, \sigma_n$ of loops in $\mathcal{L}$ such that (i) $\sigma_1 = \sigma$, (ii) $\sigma_n = \sigma'$, *and* (iii) *for all $i = 1, \dots, n-1$, σ_i and σ_{i+1}) are overlapping.*

Overlapping loops can communicate without using external edges. Hence, it is possible to pass at every shared point from a loop to another, eventually using up all the edges as many times as needed. Here, we describe the connection algorithm.

Definition 1.2.9. Composition Algorithm

Consider a natural linear combination of loops $\mathcal{T} = \{(\sigma_1, c_1) \dots (\sigma_l, c_l)\}$. Then we can construct a loop σ as a composition of $\mathcal{T}$ with the following iterative algorithm. The algorithm lasts l steps, at every step i we construct a temporary loop σ'_i. At the end of the algorithm, the sought loop is $\sigma = \sigma'_l$. At every step i we also construct a set of loops $\mathcal{L}_i = \{\sigma_{j_1}, \dots \sigma_{j_i}\}$ such that σ'_i is a composition of $\mathcal{L}_i$ with coefficients $c_{j_1}, \dots, c_{j_i}$.

1. *As a first step, we have $\mathcal{L}_1 = \{\sigma_1\}$ and $\sigma'_1 = \sigma_1^{c_1}$.*
2. *For all steps $i > 1$, we determine a loop $\sigma_{j_{i+1}} \in \mathcal{L} \setminus \mathcal{L}_i$ such that the set $\mathcal{L}_{i+1} = \mathcal{L}_i \cup \{\sigma_{j_{i+1}}\}$ is overlapping. We determine such a loop, by scanning all loops in $\mathcal{L} \setminus \mathcal{L}_i$, till we find one that share a node with $\mathcal{L}_{i+1}$. Such a loop necessarily exists else we have that there is no node in common between $\mathcal{L}_i$ and $\mathcal{L} \setminus \mathcal{L}_i$, which is impossible since $\mathcal{L}$ is overlapping. Let v be the node in common between $\mathcal{L}_i$ and $\sigma_{j_{i+1}}$. Since σ'_i is a composition of $\mathcal{L}_i$, we have that it passes through v. Hence we can write σ'_i as a loop starting from v and ending in v: $\sigma'_i = vav_1a_1 \dots v_m a_m v$. We can do the same with $\sigma_{j_{i+1}} = vav'_1a'_1 \dots v'_m a'_m v$. Hence, we can set $\sigma'_{i+1} = \sigma'_i \cdot (\sigma_{j_{i+1}})^{c_{j_{i+1}}}$. For each edge e, we have $|\sigma'_{i+1}|_e = |\sigma'_i|_e + c_{j_{i+1}}|\sigma_{j_{i+1}}|_e = \sum_{i=h}^{i+1} c_{j_h}|\sigma_{j_h}|_e$. Hence, the iteration invariant holds.*

Since overlapping loops allow the construction of more complex ones, they are particularly important in the construction of infinite paths. Here, we prove that the overlapping of a set of loops is equivalent to the strong connection property of the subgraph containing only their edges.

Definition 1.2.10. Induced Subgraph

Given a set of loops $\mathcal{L}$ in G, the subgraph induced by $\mathcal{L}$ is $G' = (V', E')$, where V' and E' are all and only the nodes and the edges, respectively, belonging to a loop in $\mathcal{L}$.

Lemma 1.2.2. *Overlap Lemma*

Let G be a graph, $\mathcal{L}$ be a set of loops in G, and $G' = (V', E')$ be the subgraph of G induced by $\mathcal{L}$, then the following statements are equivalent:

1. *$\mathcal{L}$ is* overlapping.
2. *The subgraph G' is strongly connected.*
3. *There exists $u \in V'$ such that for all $v \in V'$ there exists a path in G' from u to v.*

Proof. [$1 \Rightarrow 2$] If $\mathcal{L}$ is overlapping, then, for all pairs of loops σ_1, σ_2, there exists a sequence of loops that links σ_1 with σ_2. Thus, from any node of σ_1, it is possible to reach any node of σ_2. Hence, G' is strongly connected.

[$2 \Rightarrow 3$] Trivial.

[$3 \Rightarrow 2$] Let $u \in V'$ be a witness for (3). Let $v, w \in V'$, we prove that there is a path from v to w. We have that u is connected to both v and w. Since all edges in G' belong to a loop, for all edges $(u', \cdot, v')$ along the path from u to v there is a path from v' to u'. If in G' there exists a node u such that there is a path from u to all other nodes v of G', since u and v are connected by a chain c of overlapping loops, the path from u to v passes through the nodes that connect any pair of adjacent loops in c. Since any two such nodes are connected by a path in a loop, there exists a reverse path in the same loop. Thus we can construct a path from v to u by using the reverse paths from the nodes that connect the pair of adjacent loops in the chain c. Since this holds for all v, then for all $v, w \in V'$ we can construct a path from v to w by passing trough u.

[$2 \Rightarrow 1$] If G' is strongly connected, for all $\sigma_1, \sigma_2 \in \mathcal{L}$ there is a path ρ in G' from any node of σ_1 to any node of σ_2. This fact holds since G' is induced by $\mathcal{L}$, so ρ uses only edges of the loops in $\mathcal{L}$. While traversing ρ, every time we move from one loop to the next, these two loops must share a node. Therefore, all pairs of adjacent loops used in ρ are overlapping. Thus $\mathcal{L}$ is overlapping. □

The above lemma implies that if $\mathcal{L}$ is overlapping then it is also connected, since G' is strongly connected.

...

...

2
Fair Scheduling

Contents

2.1 Fairness Goals

In this thesis we study fairness goals stronger than those definable through the most renowned temporal logics for discrete-time and completely-determined models. These goals' aim is to impose that a scheduler dedicates the same amount of resources to each task.

Along a system's run, the compliance to an equal distribution of resources, can be monitored by observing for each pair of activities, how the difference between their occurrences grows along the prefixes. If the module of a difference grows too much, this is symptom that one activity is executed less often than the other. Ideally this difference should be 0, however, this is impossible because after executing the first activity the difference is already no longer 0. Moreover the precedence graph may be structured in a such a way, that it needs to execute an activity a lot of times in a row before being able to switch to another one, during that time the difference inevitably grows. Hence, our aim is to keep this difference as small as possible and bounded by some function on the length of the prefix. The least growing function is the constant one and it provides our strongest fairness condition.

Definition 2.1.1. *A color sequence $x \in [k]^\omega$ is said bounded (in difference) by a constant $C \in N$ if for every pair of colors $i, j \in [k]$ and every prefix $x^{\leq n}$, the module of the difference between their number of occurrences $d_{\mathit{iff}\,i,j}(x^{\leq n})$ on that prefix is bounded by C. A path ρ in a k-colored graph is said bounded (in difference) by $C \in N$ if its color sequence is bounded by C. A path or color sequence is said bounded, when it is bounded by some constant.*

Example 2.1.1. *The color sequence $(0011)^\omega \in [1]^\omega$ is bounded by $C = 2$.*

When a path is not bounded, we can still have a weaker form of fairness by bounding the difference of activities with some other function on the length of the prefix. However, if this function grows linearly like $C \cdot n$, then on infinitely many prefixes we have $d_{\mathit{iff}\,i,j}(x^{\leq n}) = C \cdot n$. Hence, the share of resources dedicated to activity i (i.e. $\frac{|x^{\leq n}|_i}{n}$)is greater than the share of resources dedicated to activity j (i.e. $\frac{|x^{\leq n}|_i}{n}$) by the value C, obviously this is no fairness. When the function grows sublinearly, like $g(n) \in o(n)$, the difference between the share of resources dedicated to two activities is not greater than $\frac{g(n)}{n}$. Since this value tends to zero for n that tends at infinity, we have that it becomes a negligible difference in an infinite run. Hence, when the difference between the occurrence of two activities grows sublinearly, the path may be considered as asymptotically fair.

Definition 2.1.2. *A color sequence $x \in [k]^\omega$ is said balanced if for every pair of colors $i, j \in [k]$, the difference between their number of occurrences grows sublinearly on the length of the prefix, i.e. $\lim_{n\to+\infty} \frac{d_{\mathit{iff}\,i,j}(x^{\leq n})}{n} = 0$. A path ρ in a k-colored graph is said balanced if its color sequence is balanced*

Example 2.1.2. *The color sequence $x = \prod_{i=1}^{+\infty}((1100)^i 11) \in [1]^\omega$ is balanced but not bounded. It is easy to see that every time a piece in the infinite concatenation ends the color difference is increased by 2, i.e., $d_{\mathit{iff}\,1,0}(\prod_{i=1}^{n+1}((1100)^i 11)) = d_{\mathit{iff}\,1,0}(\prod_{i=1}^{n}((1100)^i 11)) + 2$. Hence, it holds that $d_{\mathit{iff}\,1,0}(\prod_{i=1}^{n}((1100)^i 11)) = 2n$ and the path is not bounded. Since $|\prod_{i=1}^{n}((1100)^i 11)| = \sum_{i=1}^{n}(4i+2) = 2n(n+2)$, we have that at every step $2n(n+2)$ where a piece of the infinite*

concatenation ends, the length of the prefix grows asymptotically faster than the difference of colors. Since in the intermediate prefixes $x^{\leq m}$ *with* $m \in [2n(n+2), 2(n+1)(n+3)]$*, the color difference oscillates in* $[2n, 2n+2]$ *we have that* $\lim_{m\to+\infty} \frac{d_{iff\,1,0}(x^{\leq m})}{m} = 0$.

One may define other intermediate notions of fairness between the two defined above, by explicitly requiring that the difference of occurrences of activities does not grow asymptotically faster than a given sublinear function. However, we leave such a discussion to Section 3.6.

From the fair paths' property defined, we can construct two colored winning conditions.

Definition 2.1.3. *The bounded difference (resp. balance) goal* W_{bd} *(resp.* W_{bl}*)is the set of all bounded (resp. balanced) infinite color sequences on* $[k]$.

2.2 Graphs Characterization

In this section, we characterize the existence of a balanced or bounded path by means of properties of the underlaying colored graph. In particular, we observe that we can compose simple loops for the construction of a fair path. On the converse, we prove that a fair path always contain some regularities, that allow us to extract loops components. By evaluating the properties of such loops, we show that whenever there exist of a fair path, a set of simple loops can be used to construct some regular fair path.

2.2.1 The Bounded Difference Problem

The bounded difference property states that along the prefixes of a path, the difference between the occurrences of two colors is bounded by some constant. Since the graph has a finite number of nodes, and since each difference can assume at most a finite number of values, in a bounded path, along the prefixes, there is eventually a repetition of the same ending node and the same difference vector. When a repetition is found, we determine a loop which did not induce any change in the difference of colors. Since the difference is additive, the loop has a null difference vector, i.e., all colors occur the same number of times on it.

Definition 2.2.1. *A loop* σ *is said perfectly balanced if all colors occur the same number of times on it, i.e.,* $\underline{d_{iff}}(\sigma) = \mathbf{0}$.

It is immediate to see that a perfectly balanced loop σ allows as to construct a bounded path σ^ω. Hence, by our previous argument we can easily prove the following lemma.

Lemma 2.2.1. *Given a graph G, the following statements are equivalent:*

1. *There exists a bounded difference path.*
2. *There exists a periodic bounded difference path.*
3. *There exists a perfectly balanced loop.*

Proof. 1. $[1 \to 3]$ If $\rho = v_0 a_0 \dots v_n a_n \dots$ is an infinite bounded difference path, then there exists a constant C such that the absolute value of all color differences is smaller than C. Since both the set of nodes and the possible difference vectors along ρ are finite, we can find two indexes $i < j$ such that $v_i = v_j$ and $\underline{d_{iff}}(\rho^{\leq i}) = \underline{d_{iff}}(\rho^{\leq j})$. So, $\sigma' = v_i a_i v_{i+1} a_{i+1} \dots v_{j-1} a_{j-1} v_j$ is a perfectly balanced loop.

2. $[3 \to 2]$ If there exists a perfectly balanced loop σ, then σ^ω is a periodic bounded difference path. Indeed, every time σ closes the color gap vector is zero, so each color gap in module never increases beyond the maximum color gap in a subpath of σ.

3. $[2 \to 1]$ Trivial

□

The lemma relates the existence of a bounded path to the existence of a perfectly balanced loop. At this point, obtaining a characterization in terms of simple loops is just a matter of decomposing the balanced loop. To this aim we define a property of a set of loops, which states that, by using a certain number of times each loop, all the colors occur the same number of times if the loops were composed together.

Definition 2.2.2. Difference Vecto of n.l.c.

Let G be a k-colored graph. Given a set of loops $\mathcal{L} = \{\sigma_1, \dots, \sigma_l\}$ the difference value of a natural linear combination of $\mathcal{L}$ with respect to non-null coefficients $c_1, \dots, c_l \in \mathbb{N}_{-0}$, is the difference vector $\sum_{i=1}^{l} c_i \cdot \overline{d_{iff}}(\sigma_i)$.

In this definition, we are using the color k as a reference for all other colors. Indeed, whenever all differences between the occurrences of a color and the occurrences of color k are null, then we can already state that all colors occur the same number of times. At this point,using the decomposition and composition lemmas presented in the preliminary section on graphs we obtain the following result.

Lemma 2.2.2. *Let G be a k-colored graph. There exists a perfectly balanced loop in G if and only if there exists an overlapping set $\mathcal{L}$ of simple loops of G, with an n.l.c. of difference value equal to zero.*

Proof. [only if] If there exists a perfectly balanced loop σ, by Lemma 1.2.1 the loop is the composition of a tuple $\mathcal{T}$ of simple loops. Let $\mathcal{L}$ be the set of distinct loops occurring in $\mathcal{T}$, and for all $\sigma' \in \mathcal{L}$, let $c_{\sigma'}$ be the number of times σ' occurs in $\mathcal{T}$. Since in the computation of the difference vector of a path it does not matter the order in which the edges are considered, we have $\sum_{\sigma' \in \mathcal{L}} c_{\sigma'} \cdot \overline{d_{iff}}_k(\sigma') = \overline{d_{iff}}_k(\sigma) = \mathbf{0}$. Finally, since the loops in $\mathcal{L}$ come from the decomposition of a single loop σ, we have that $\mathcal{L}$ is overlapping.

[if] Let $\mathcal{L} = \{\sigma_1, \dots, \sigma_l\}$ be an overlapping set of simple loops such that $\sum_{i=1}^{l} c_i \cdot d_{iff}(\sigma_i) = \mathbf{0}$. By using the algorithm presented in Definition 1.2.9, we construct a loop σ which is a linear composition of $\mathcal{L}$ with coefficients $c_1, \dots, c_l$. Then, we have that σ is perfectly balanced, since $\overline{d_{iff}}_k(\sigma) = \sum_{i=1}^{l} c_i \cdot \overline{d_{iff}}_k(\sigma_i) = 0$.

□

The following theorem is a direct consequence of the previous two lemmas.

Theorem 2.2.1. *A graph G satisfies the bounded difference problem if and only if there exists an overlapping set $\mathcal{L}$ of simple loops of G, with an n.l.c. of difference value equal to zero.*

2.2.2 The Balance Problem

The balance property states that along the prefixes of a path, the difference between the occurrences of two given colors grows asymptotically slower than any function linear in length of the prefix, and, hence, the difference is negligible in the long run. Due to the limit in the speed of increment of the difference vector, in a given interval of lengths the differences are bounded by some constant that does not grow linearly with the size of the lengths. Is it possible that during these intervals, where the number of nodes and value of differences is limited, there is some unavoidable periodicity, like in the case of the bounded problem? The answer lays in the necessity for sequences of vectors of integer numbers to grow polynomially whenever a repetition is not possible. Hence, the answer to the question is positive, as the next lemma shows that no periodicity implies a linear increment of a component of a sequence of vectors.

Definition 2.2.3. Linear Combination

Let $A = (a_1, \ldots, a_n) \subseteq \mathbb{Z}^d$ be a finite set of vectors, the linear combination of A with coefficients $c_1, \ldots, c_l$ is them vector $\sum_{i=1}^{n} c_i \cdot a_i$.

Lemma 2.2.3. *Let $A \subset \mathbb{Z}^d$ be a finite set of vectors such that there is no subset $A' \subseteq A$ with an n.l.c. of value zero. Let $\{(a_{n,1}, \ldots, a_{n,d})\}_{n\in\mathbb{N}}$ be an infinite sequence of elements of A, and $S_{n,i} = \sum_{j=0}^{n} a_{j,i}$ be the partial sum of the i-th component, for all $n \in \mathbb{N}$ and $i \in [d]$. Then, there exists at least an index h such that $\lim_{n\to\infty} \frac{S_{n,h}}{n} \neq 0$.*

The proof of Lemma 2.2.3, makes use of the following lemma which allows to compute the coefficients of an integer linear combination, knowing the vectors involved.

Lemma 2.2.4. *Let $Ax = \mathbf{0}$ be a linear homogeneous system with $A \in \mathbb{Q}^{n\times m}$. If the system has a solution x such that $x \geq \mathbf{0}$ and $\sum_{i=1}^{m} x_i = 1$, then it has a solution with all natural components, and at least one strictly positive component.*

Proof. Let S be the set containing all and only the solutions x of $Ax = \mathbf{0}$, with all non-negative components and such that $\sum_{i=1}^{m} x_i = 1$. By hypothesis, S is not empty. Let $A' = (\frac{A}{1,1,\ldots,1})$ and $b' = (\frac{\mathbf{0}}{1})$, we have $S = \{x \in \mathbb{R}^m \mid A'x = b', x \geq \mathbf{0}\}$. By a well known result in linear programming (see, for instance, Theorem 3.5 of [NW88]), S contains a *basic* solution, i.e., there exists a non-singular submatrix $C \in \mathbb{R}^{r\times r}$ of A', given by the columns $i_1, \ldots, i_r$ and the rows $j_1, \ldots, j_r$ of A', such that there is a point $(z_1, \ldots, z_m) \in S$ such that $z' = (z_{i_1}, \ldots, z_{i_r})$ is the unique solution to the system of linear equations $Cz' = b$ where $b = (b'_{j_1}, \ldots, b'_{j_r})^T$, and for all $i \notin \{i_1, \ldots, i_r\}$, $z_i = 0$. By Cramer's theorem, for all $k \in [r]$, we have $z_{i_k} = \frac{det(C'_{i_k})}{det(C')}$ where C'_{i_k} is the matrix obtained from C' by replacing the i_k-th column with the column vector b. Since the determinant of a rational matrix is rational, z is a point of S with all rational coefficients, i.e., z is a solution of $Ax = \mathbf{0}$ with all non-negative rational coefficients such that $\sum_{i=1}^{m} z_i = 1$. Clearly, z has at least one positive coefficient. Now, since the system $Ax = \mathbf{0}$ is homogeneous, by multiplying each component of z by the least common denominator of all components, we obtain the thesis. □

Now we are ready to prove Lemma 2.2.3.

Proof. Let $A = \{(x_{1,1}, \ldots, x_{1,d}), \ldots, (x_{m,1}, \ldots, x_{m,d})\}$ and consider the function $f : \mathbb{R}^m \mapsto \mathbb{R}_+$ such that $f(c_1, \ldots, c_m) = \max_{1 \leq i \leq d}\{|\sum_{j=1}^m c_j \cdot x_{j,i}|\}$. By construction, f is a continuous function. Let now $K \subset \mathbb{R}^m$ be the set $\{(c_1, \ldots, c_m) \in [0,1]^m \mid \sum_{i=1}^m c_i = 1\}$. Note that $\mathbf{0} \notin K$ and that K is compact, since it is a finite dimensional space defined by a linear equation. Hence, by Weierstrass' Theorem, f admits a minimum value M Now, since A is a set of convexly independent vectors, M must be strictly positive. Indeed, if by contradiction $M = 0$, there is a non-null vector $(c_1, \ldots, c_m) \in K$ such that $\sum_{j=1}^m c_j \cdot x_{j,1} = \ldots = \sum_{j=1}^m c_j \cdot x_{j,d} = M = 0$. By Lemma 2.2.4 the system $\sum_{j=1}^m c_j \cdot x_{j,i} = 0$ has a natural solution c with at least one positive component. This solution gives rise to an n.l.c. of some vectors of A (those corresponding to positive components of c) with value zero, which contradicts the hypotheses on A.

Then, consider the sequence $\{(a_{n,1}, \ldots, a_{n,d})\}_n$ and its partial sums $S_{n,i} = \sum_{j=0}^n a_{j,i}$. Moreover, let $\delta_{i,n}$ be the number of times for which the vector $(x_{i,1}, \ldots, x_{i,d})$ occurs in the previous sequence up to position n and let $c_{i,n} = \delta_{i,n}/n$. Then, $(S_{n,1}, \ldots, S_{n,d}) = \sum_{i=1}^m \delta_{i,n} \cdot (x_{i,1}, \ldots, x_{i,d}) = n \cdot \sum_{i=1}^m c_{i,n} \cdot (x_{i,1}, \ldots, x_{i,d})$. Since we have $\sum_{i=1}^m \delta_{i,n} = n$ for all $n \in \mathbb{N}$, it is obvious that $(c_{1,n}, \ldots, c_{m,n}) \in K$. By the convex independence hypothesis on A, it holds that for all $n \in \mathbb{N}$ there exists at least an index i, with $1 \leq i \leq d$, such that $S_{n,i} \neq 0$. Let $\{j_n\}_n$ be an index sequence such that $|S_{n,j_n}| = \max_{1 \leq i \leq d}\{|S_{n,i}|\} > 0$, for all $n \in \mathbb{N}$. Since $\{j_n\}_n$ can assume at most d different values, there exists a value h which occurs infinitely often in it. Let $\{h_i\}_i$ be the index sequence such that $j_{h_i} = h$ and there is no $l \in]h_i, h_{i+1}[$ with $j_l = h$. Then, from $\{S_{n,h}\}_n$ we can construct the extracted sequence $\{S_{h_i,h}\}_i$. Now, we have that $|S_{h_i,h}| = \max_{1 \leq j \leq d}\{|S_{h_i,j}|\} = h_i \cdot \max_{1 \leq j \leq d}\{|\sum_{k=1}^m c_{k,h_i} \cdot x_{k,j}|\} = h_i \cdot f(c_{1,h_i}, \ldots, c_{1,h_i}) \geq h_i \cdot M > 0$. Hence, $\lim_{i \to \infty} \frac{|S_{h_i,h}|}{h_i} \geq M > 0$. Since $\left\{\frac{|S_{h_i,h}|}{h_i}\right\}_i$ is an extracted sequence of $\left\{\frac{|S_{n,h}|}{n}\right\}_n$, we finally obtain that $\lim_{n \to \infty} \frac{S_{n,h}}{n} \neq 0$. □

The lemma proves that whenever we add incrementally a sequence of vectors that cannot balance themselves, then at least one component does not grow sublinearly. Now, consider an infinite path, for each prefix it is possible to consider the path quasi-segmented in an infinite sequence of loops given by the incremental quasi-segmentation of its prefixes. Each loop is associated with a difference vector, and every time a loop ends, the difference vector of the prefix is the sum of the vectors of the loops met so far. Hence, if the loops cannot balance themselves with a natural linear combination, the path cannot be balanced.

Lemma 2.2.5. *In a k-colored graph G there exists a balanced path only if there exists a connected set $\mathcal{L}$ of simple loops of G, with an n.l.c. of difference value equal to zero.*

Proof. If there exists an infinite balanced path ρ, since the set of nodes is finite, there is a set V' of nodes occurring infinitely often in ρ. Let ρ' be a suffix of ρ containing only nodes in V'. The path ρ' is balanced and it is composed by an infinite sequence of simple loops on V'. (see Definition 1.2.6 for further details). Let $\mathcal{L}$ be the (finite) set of such simple loops, and let $A \subset \mathbb{Z}^{k-1}$ be the set of difference vectors of the loops in $\mathcal{L}$. Every time a loop trough V' closes along ρ', the difference vector up to that point is the sum of the difference vectors of the simple loops occurred so far, plus the difference vector of the remaining simple path. Since the remaining simple path

cannot have length greater than $|V'|$, the difference vector up to that point differs from a sum of a sequence of elements of A by a constant-bounded term. Let $n(j)$ be the index of the j-th point where a loop is closed along ρ'. Since ρ' is balanced, each component of the difference sequence $\{d_{iff}(\rho'^{\leq n(j)})\}_{i\in\mathbb{N}}$ is in $o(j)$. By Lemma 2.2.3, this is possible only if A has a subset A' with an n.l.c. of value zero. Thus, the set of loops $\mathcal{L}'$ with difference vectors in A' has an n.l.c with difference value zero. Moreover, since the loops in $\mathcal{L}'$ are constructed with edges of ρ', they are connected. This concludes the proof. □

We determined that a periodicity among the simple loops is necessary for the existence of a balanced path. Is it sufficient as well? Provided that a set of loops is connected and admits a natural linear combination with difference value zero, we can construct a balanced path. The key to the construction is to use all the loops periodically as many times as multiples of the relative coefficient of the nl.c.. However, moving from one loop to another requires the use of the connecting paths that add a non-balanced contribution to the number of occurrences. The solution is to add such connecting paths less and less often with respect to the use of the main loops. In such a way, their unbalanced contribution grows sublinearly and is negligible asymptotically. This intuition is formalized in the following lemma.

Lemma 2.2.6. *In a k-colored graph G if there exists a connected set $\mathcal{L}$ of simple loops of G, with an n.l.c. of difference value zero then there exists a balanced path.*

Proof. Let $\mathcal{L} = \{\sigma_0, \dots, \sigma_{l-1}\}$ be a connected set of simple loops having an n.l.c. with difference value zero, and coefficients $c_0, \dots, c_{l-1}$. For all $i = 0, \dots, l-1$, let v_i be the initial node of σ_i. Since $\mathcal{L}$ is connected, there exists a path ρ_i from v_i to $v_{(i+1) \bmod l}$. For all $j > 0$, define the loop $\pi_j = \sigma_0^{j\cdot c_0}\rho_0\sigma_1^{j\cdot c_1}\rho_1 \dots \sigma_{l-1}^{j\cdot c_{l-1}}\rho_{l-1}$. We claim that the infinite path $\pi = \prod_{j>0}\pi_j$ is balanced. Each time a π_j block ends along π, the part of the difference vector produced by the loops of $\mathcal{L}$ is zero. So, when a π_j ends, the difference vector is due only to the paths ρ_i. Since the index of the step $k(j)$ at which π_j ends grows quadratically in j and the difference vector $d_{iff}(\pi_1 \dots \pi_j)$ grows linearly in j, we have that $\lim_{j\to\infty} d_{iff}(\pi_1 \dots \pi_j)/k(j) = \mathbf{0}$. It can be shown that in the steps between $k(j)$ and $k(j+1)$, the i-th component of the difference vector differs from the one of $d_{iff}(\pi_1 \dots \pi_j)$ no more than a function $C_{i,j}$ that grows linearly in j. Specifically, $C_{i,j} = MP_i + jMA_i$, where MP_i is the sum, for all ρ_j, of the maximum modulus of the i-th component of the difference vector along ρ_j, and MA_i is the sum, for all σ_j, of the maximum modulus of the i-th component of the difference vector along σ_j. As a consequence, $\lim_{k\to\infty} d_{iff}(\pi^{\leq k})/k = \mathbf{0}$ and π is balanced. □

By merging together the two previous lemmas, as a corollary, we obtain the graph theoretic characterization for the balance problem.

Theorem 2.2.2. *A graph G satisfies the balance problem if and only if there exists a connected set $\mathcal{L}$ of simple loops of G, with an n.l.c. of difference value equal to zero.*

This theorem, alongside proof of Lemma 2.2.6, shows that a connect set of simple loops having zero as difference value of an n.l.c. constitute a representation of a balanced path.

2.2.3 2-Colored Graphs

From the preliminary section, we know that a bounded path is also balanced and that a balanced path is not necessarily bound. However, is it possible that whenever there exists a balanced path we can also find a bounded path in the same graph? Or stated in different terms, does the existence of a connected set of simple loops with zero as difference value of an n.l.c. imply the existence of an overlapping set of simple loops with zero as difference value of an.l.c.? In general, the answer to this question is negative as shown in the following example.

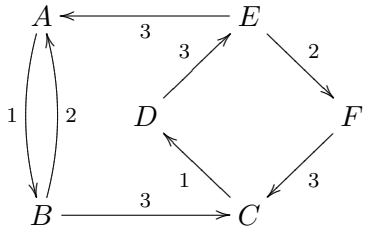


Figure 2.1: A 3-colored graph satisfying the balance problem, but not the bounded difference problem.

Example 2.2.1. *Consider the graph G in Fig. 2.1. First note that, up to rotation, there are just three simple loops in it: $\sigma_1 = A \cdot B \cdot A$, $\sigma_2 = C \cdot D \cdot E \cdot F \cdot C$, and $\sigma_3 = A \cdot B \cdot C \cdot D \cdot E \cdot A$. It is easy to see that $d_{iff}(\sigma_1) = (1, 1)$, $d_{iff}(\sigma_2) = (-1, -1)$, and $d_{iff}(\sigma_3) = (-1, -3)$. On one hand, since the connected set of simple loops $\{\sigma_1, \sigma_2\}$ has an n.l.c. with difference value zero, we obtain that there is a balanced path in G. On the other hand, for all the three overlapping sets of loops ($\{\sigma_1, \sigma_3\}$, $\{\sigma_2, \sigma_3\}$, and $\{\sigma_1, \sigma_2, \sigma_3\}$) there is no way to obtain a zero difference value of an n.l.c. with all coefficients different from zero. So, there is no bounded difference path in G.*

Hence, the bounded and balance problems are not equivalent in general. However, when we deal with 2-colored graph, the color difference vector becomes a natural number. So, if $\mathcal{L}$ is a connected set of simple loops having zero as difference value of an n.l.c., then there must be either a perfectly balanced simple loop or two loops with difference vectors of opposite sign. Notice that two loops σ, σ' with color differences of opposite sign have the following n.l.c. of difference value zero: $|d_{iff}(\sigma')| \cdot d_{iff}(\sigma) + |d_{iff}(\sigma)| \cdot d_{iff}(\sigma') = 0$. If the two loops are connected but not overlapping, we can construct a sequence of adjacent overlapping simple loops connecting them. In this sequence, we are always able to find a perfectly balanced simple loop or two overlapping simple loops with difference vectors of opposite sign. Therefore, on a 2-colored graph the balanced and bounded problems coincide, as proved in more details in the following lemma.

Lemma 2.2.7. *Let G be a 2-colored graph. If there exists a connected set of simple loops of G with zero as difference value of an n.l.c., then there exists an overlapping set of simple loops of G with zero as difference value of an n.l.c.*

Proof. In a 2-colored graph, the difference vector of any path ρ is simply an integer. Let $\mathcal{L}$ be a connected set of simple loops with zero as a difference value of an n.l.c.. If $\mathcal{L}$ contains a simple loop σ such that $d_{iff}(\sigma) = 0$, then $\{\sigma\}$ is trivially an overlapping set.

If $\mathcal{L}$ contains no perfectly balanced loop, then all the difference vectors of the loops in $\mathcal{L}$ cannot have the same sign, otherwise it is not possible to have a non-trivial natural combination $\sum_{\sigma\in\mathcal{L}} c_\sigma d_{iff}(\sigma) = 0$.

Thus, let $d_{iff}(\sigma) > 0$ and $d_{iff}(\sigma') < 0$, for $\sigma, \sigma' \in \mathcal{L}$. If σ and σ' are overlapping, then $\{\sigma, \sigma'\}$ is the overlapping set we are looking for. If σ and σ' are not overlapping, since they are connected, there exist a path ρ_1 from σ to σ' and a path ρ_2 from σ' to σ. σ So, there exist four indexes i, i', j, j' such that $\rho_1(i)$ is the last node of ρ_1 in σ, $\rho_1(j)$ is the first node of ρ_1 in σ', $\rho_2(i')$ is the last node of ρ_2 in σ', and $\rho_2(j')$ is the first node of ρ_2 in σ. Then, within the loop σ there exists a simple path ρ from $\rho_2(j')$ to $\rho_1(i)$ and, within the loop σ', there exists a simple path ρ' from $\rho_1(j)$ to $\rho_2(i')$. We then set $\rho'_1 = \rho_1(i) \dots \rho_1(j)$ and $\rho'_2 = \rho_2(i'), \dots, \rho_2(j')$. We observe that the pairs of paths (ρ, ρ'_1), (ρ', ρ'_1), (ρ', ρ'_2), and (ρ, ρ'_1) have only one node in common. Moreover, ρ and ρ' have no node in common since σ and σ' are not overlapping. So, the loop $\sigma'' = \rho\rho'_1\rho'\rho'_2$ is not simple if and only if ρ'_1 and ρ'_2 have a node in common. Now, observe that two loops π_1 and π_2 with difference vectors of opposite sign have zero as difference value of an n.l.c. with coefficients $|d_{iff}(\pi_2)|$ and $|d_{iff}(\pi_1)|$, since $|d_{iff}(\pi_2)|d_{iff}(\pi_1) + |d_{iff}(\pi_1)|d_{iff}(\pi_2) = 0$. We conclude with the following case analysis.

1. If σ'' is simple, then (σ, σ'') and (σ', σ'') are pairs of overlapping sets.

   (a) If $d_{iff}(\sigma'') = 0$ then $\{\sigma''\}$ is an overlapping set having zero as an n.l.c.

   (b) If $d_{iff}(\sigma'') > 0$ then $\{\sigma', \sigma''\}$ is an overlapping set having zero as an n.l.c.

   (c) If $d_{iff}(\sigma'') < 0$ then $\{\sigma, \sigma''\}$ is an overlapping set having zero as an n.l.c.

2. If ρ'_1 and ρ'_2 have nodes in common, there exist two indexes k, k' such that it holds $\rho'_1(k) = \rho'_2(k')$. So, we can construct two loops $\sigma'_1 = \rho\rho'_1(1) \dots \rho'_1(k) \dots \rho'_2(|\rho'_2|)$ and $\sigma'_2 = \rho'\rho'_2(1) \dots \rho'_2(k') \dots \rho'_1(|\rho'_1|)$.

   (a) If $d_{iff}(\sigma'_i) = 0$, for some $i \in \{0, 1\}$, then $\{\sigma'_i\}$ is an overlapping set having zero as an n.l.c.

   (b) If $d_{iff}(\sigma'_2) > 0$ then $\{\sigma', \sigma'_2\}$ is an overlapping set having zero as an n.l.c.

   (c) If $d_{iff}(\sigma'_1) < 0$ then $\{\sigma, \sigma'_1\}$ is an overlapping set having zero as an n.l.c.

   (d) If $d_{iff}(\sigma'_1) > 0$ and $d_{iff}(\sigma'_2) < 0$, then $\{\sigma'_1, \sigma'_2\}$ is an overlapping set having zero as an n.l.c.

□

Due to the above characterization, both decision problems can be solved efficiently, by using a Bellman-Ford algorithm to find two loops of opposite color difference sign, if such exist.

Theorem 2.2.3. *A 2-colored graph $G = (V, E)$ satisfies the bounded difference problem if and only if it satisfies the balance problem. Both problems can be solved in time $\mathcal{O}(|V| \cdot |E|)$.*

Proof. The algorithm is the following: at first we use a deep-first search to construct a simple loop. If the loop is null then it is the perfectly balanced loop and the algorithm terminates. If the loop is positive (resp. negative) we first decompose the graph in its connected component by using Tarjan's algorithm in time $\mathcal{O}(|V| + |E|)$. Then on each connected component we apply Bellman-Ford algorithm for single-source maximum (resp. minimum) paths, starting at any node. Bellman-Ford algorithm finds a negative (resp. positive) loop if one exists. If one run of the algorithm, on a connected component, finds a negative (resp. positive) loop then the balance and bounded problem are solved with positive answer. If no run of the algorithm finds a negative (resp. positive) loop, then the problems are solved with negative answer. Each run of the Bellman-Ford algorithm takes time $\mathcal{O}(|V'| \cdot |E'|)$ where V' and E' are the set of nodes and the set of edges of the connected component. So if S is the set of connected components G', we have the whole algorithm runs in time $\mathcal{O}(|V| + |E| + \sum_{G' \in S} |V_{G'}| \cdot |E_{G'}|) \in \mathcal{O}(\sum_{G' \in S} |V_{G'}| \cdot |E|) = \mathcal{O}(|E| \sum_{G' \in S} |V_{G'}|) = \mathcal{O}(|E| \cdot |V|)$. □

2.3 Solving the Balance Problem

In order to solve the balance problem it is sufficient to determine a set of connected simple loops having zero as difference value of an n.l.c. Indeed, such loops can be used to construct a balanced path as shown in the proof of Lemma 2.2.6. Since simple loops are connected if and only if they belong to the same connected component, we can restrict our investigation only to such connected components. Hence, in the following, we make the assumption that the underlaying graph is strongly connected and we look for a set of loops with zero as difference value of a natural linear combination. To this purpose we use techniques developed for the solutions of flow problems in operative research [HL03].

Consider a strongly connected k-colored graph $G = (V, E)$, our purpose is to relate edges forming loops to an n.l.c. of difference value zero. Hence, our formalization needs to *(i)* highlight edges as many time as they are needed in the linear combination *(ii)* demand that the edges contain the same number of colors *(iii)* demand that all the edges are used in a finite number of loops. The first aim is achieved by associating to each edge e a flow variable x_e representing how many times the edge is used in the linear combination. For the second point, we simply need to impose that the flow passing through each color is the same for every color: $\sum_{e \in E(a)} x_e = \sum_{e \in E(k)} x_e$ for all $a \in [k-1]$. The third point requires some more explanation, however, if the edges form loops, then for each edge entering in a node there must be an edge exiting from that node: $\sum_{e \in v^{\leftarrow}} x_e = \sum_{e \in v^{\rightarrow}} x_e$. This necessary condition turns out to be also a sufficient one as shown in the following lemma.

Lemma 2.3.1. *Given a graph $G = (V, E)$, there exists a vector $x_e : E \to \mathbb{N}$ such that for each edge $v \in V$ it holds $\sum_{e \in v^{\leftarrow}} x_e = \sum_{e \in v^{\rightarrow}} x_e$ if and only if there exists a sequence of simple loops $\mathcal{T} = \sigma_1, \ldots, \sigma_l$ such that $x_e = \sum_{i=1}^{l} |\sigma_i|_e$.*

if. Let $\mathcal{T} = \sigma_1, \ldots, \sigma_l$ be a sequence of simple loops and let $x_e = \sum_{i=1}^{l} |\sigma_i|_e$. Consider a node $v \in V$, and let $y_{v,i} \in \{0, 1\}$ be a value indicating whether the node v is used in the loop σ_i. Precisely $y_{v,i} = 1$ if and only if the loop σ_i passes through v. Since each loop passing through v

contains only one edge entering in v and only one edge exiting from v, and since a loop non-passing through v contains none of the above, we have for each v, $\sum_{e \in v^{\rightarrow}} x_e = \sum_{i=1}^{l} y_{v,i} = \sum_{e \in v^{\leftarrow}} x_e$.

[only if] Let $x_e : E \to \mathbb{N}$ be a vector such that $\sum_{e \in v^{\leftarrow}} x_e = \sum_{e \in v^{\rightarrow}} x_e$ for all$v \in V$. We propose an algorithm that computes a sequence of loops $\mathcal{T} = \sigma_1, \ldots, \sigma_l$ such that $x_e = \sum_{i=1}^{l} |\sigma_i|_e$. The algorithm is recursive on the value of the sum $\sum_{e \in E} x_e$.

1. For the base case, let $\sum_{e \in E} x_e = 0$ then the empty sequence is the sought sequence of loops such that $x_e = \sum_{i=1}^{l} |\sigma_i|_e$.

2. Let $\sum_{e \in E} x_e > 0$. In the recursive step we construct a simple loop and we remove the edges used by this loop from the function x_e, thus obtaining a new function x'_e of lower sum. The construction is iterative and proceeds as follows:

   (a) For the invariant property the following holds: we have a simple path $\rho = e_1, \ldots, e_n$ representing the partial construction of the loop, moreover we have a vector $y_e : E \to \mathbb{N}$ such that for all e, we have $y_e = x_e - |\rho|_e$

   (b) In the first step of the iteration we choose any edge e' such that $x_{e'} > 0$ and we set $\rho = e'$ and $y_e : E \to \mathbb{N}$ as the vector such that for all $e \in E \setminus \{e'\}$ we have $y_e = x_e$ and for the edge e' we have $y_e = x_e - 1$. It is easy to see that $y_e = x_e - |\rho|_e$ for all $e \in E$. At this point ρ may be a loop, in which case the iteration ends.

   (c) Consider the simple path ρ and the function y_e at an iterative step. Let v be the ending edge of ρ: in ρ there is only one entering edge in v and no exiting edges. Hence, $\sum_{e \in v^{\leftarrow}} y_e = (\sum_{e \in v^{\leftarrow}} x_e) - 1 = (\sum_{e \in v^{\rightarrow}} x_e) - 1 = (\sum_{e \in v^{\rightarrow}} y_e) - 1$. This implies that $\sum_{e \in v^{\rightarrow}} y_e > 0$, hence we can find $e' \in v^{\rightarrow}$ such that $y_{e'} > 0$. Then, compute for the next step $\rho' = \rho \cdot e'$ and the function y'_e such that $y'_{e'} = y_{e'} - 1 = x_{e'} - |\rho|_{e'} - 1 = x_{e'} - |\rho'|_{e'}$ and $y'_e = y_e = x_e - |\rho|_e = x_e - |\rho'|_e$, for all $e \neq e'$. At this point if ρ' is a loop the iteration ends.

   (d) As the ending property we have that ρ is a simple loop and $y_e : E \to \mathbb{N}$ is such that for all e, we have $y_e = x_e - |\rho|_e$. From these properties we can also prove that $\sum_{e \in v^{\leftarrow}} y_e = \sum_{e \in v^{\rightarrow}} y_e$. For all nodes v appearing in ρ there is one exiting edge from v and one entering edge in v, hence, there is one and only one edge $e' \in v^{\leftarrow}$ and one and only one edge $e'' \in v^{\rightarrow}$ appearing in ρ. So, for all edges $e \in ()v^{\rightarrow} \cup v^{\leftarrow}) \setminus \{e', e''\}$ we have $y_e = x_e$ and for all edges $e \in \{e', e''\}$ we have $y_e = x_e - 1$. It holds that $\sum_{e \in v^{\rightarrow}} y_e = (\sum_{e \in v^{\rightarrow}} x_e) - 1 = (\sum_{e \in v^{\leftarrow}} x_e) - 1 = \sum_{e \in v^{\leftarrow}} y_e$. On the other hand, for all v not appearing in ρ for all $e \in v^{\rightarrow} \cup v^{\leftarrow}$ we have $y_e = x_e$. Hence, $\sum_{e \in v^{\rightarrow}} y_e = \sum_{e \in v^{\leftarrow}} y_e$.

   At the end of the iteration we have a simple loop ρ and a vector $y_e : E \to \mathbb{N}$ such that for all $e \in E$, we have $y_e = x_e - |\rho|_e$. At this point we can apply the recursive algorithm to the vector y_e since we have $\sum_{e \in E} y_e < \sum_{e \in E} x_e$. and $\sum_{e \in v^{\leftarrow}} y_e = \sum_{e \in v^{\rightarrow}} y_e$.

□

In the following we define a system of linear equations that allows us to compute vectors $x_e : E \to \mathbb{N}$ representing a set of simple loops. Hence, the algorithm, presented in the second part

of the proof of the previous lemma, for the computation of the simple loop, is a key component of the scheduling algorithm.

Since we defined a way to represent the edges involved in the loops, and to relate them to the existence of a natural linear combination of difference value zero we can define a linear system of constraints. To the previous discussed constraints, we add two new ones: *(i)* we ask that the number of time an edge is used is non negative, i.e., $x_e \geq 0$ for all $e \in E$, *(ii)* we have to use at least one edge in order to not find the trivial solution with all values $x_e = 0$, hence, we also ask that $\sum_{e \in E} x_e > 0$.

Definition 2.3.1. *Let $G = (V, E)$ be a k-colored graph. We call* natural balance system *for G the following system of equations on the set of variables $\{x_e \mid e \in E\}$.*

1. for all $v \in V$	$\sum_{e \in v^{\rightarrow}} x_e$	$= \sum_{e \in v^{\leftarrow}} x_e$
2. for all $a \in [k-1]$	$\sum_{e \in E(a)} x_e$	$= \sum_{e \in E(k)} x_e$
3. for all $e \in E$	x_e	≥ 0
4.	$\sum_{e \in E} x_e$	> 0
5. for all $e \in E$	x_e	$\in \mathbb{N}$.

Let $m = |E|$ and $n = |V|$, the balance system has m variables and $m + n + k$ constraints.

It is easy to prove that the above system of linear constraint is feasible if and only if there exists a set of loops with zero as a difference value of an n.l.c.

Lemma 2.3.2. *There exists a set $\mathcal{L}$ of simple loops in G with zero as a difference value of an n.l.c. if and only if the natural balance system for G is feasible.*

only if. If there exists an n.l.c. of $\mathcal{L}$ with difference value zero, let c_σ be the coefficient associated with a loop $\sigma \in \mathcal{L}$. Then, we can determine a sequence of loops $\mathcal{T} = \sigma_1, \ldots \sigma_l$ obtained by adding in arbitrary order all the loops in $\mathcal{L}$ as many times as their relative coefficient. We can construct a vector $x \in R^m$ that satisfies the balance system: $x_e = \sum_{\sigma \in \mathcal{L}} c_\sigma |\sigma|_e = \sum_{i=1}^{l} |\sigma_i|_e$. The first part of Lemma 2.3.1, shows already that x_e satisfy the first set of constraints of the natural balance systems. The third, fourth and fifth set are trivially satisfied. For the second set we need to observe that the value $\sum_{e \in E(a)} x_e$ is the sum of edges colored by a color $a \in [k]$: $\sum_{e \in E(a)} x_e = \sum_{e \in E(a)} \sum_{\sigma \in \mathcal{L}} c_\sigma |sigma|_e = \sum_{\sigma \in \mathcal{L}} c_\sigma \sum_{e \in E(a)} |\sigma|_e = \sum_{\sigma \in \mathcal{L}} c_\sigma |\sigma|_a$. Due to the natural linear combination of value zero we have that for all $a \in [k-1]$ it holds that $\sum_{\sigma \in \mathcal{L}} c_\sigma |\sigma|_a = \sum_{\sigma \in \mathcal{L}} c_\sigma |\sigma|_k$, and, hence, the second set of constraints hold.

[if] Let $x_e : E \rightarrow \mathbb{N}$ be a solution of the feasible system. By Lemma 2.3.1, we can construct a sequence of simple loops $\mathcal{T} = \sigma_1, \ldots, \sigma_l$ such that $\sum_{i=1}^{l} |\sigma_i|_e = x_e$ for all edges $e \in E$. Let $\mathcal{L}$ be the set of these loops and for each $\sigma \in \mathcal{L}$ let c_σ be the number of times σ appears in $\mathcal{T}$. Then $x_e = \sum_{\sigma \in \mathcal{L}} c_\sigma |\sigma|_e$. Due to the second set of constraints, and due to the fact that $\sum_{e \in E(a)} x_e = \sum_{\sigma \in \mathcal{L}} c_\sigma |\sigma|_a$ for all colors $a \in [k]$, we have that the natural linear combination of $\mathcal{L}$ with coefficients c_σ for each $\sigma \in \mathcal{L}$ has difference value zero.

□

So, finding a solution to the natural balance feasibility problem is equivalent to finding a set of simple loops with zero as a difference value of an n.l.c. and equivalent to finding a balanced

path. However, the feasibility of a system linear inequalities with integer variables is in general an NP-hard problem ([NW88]).On the other hand, whenever the variables are real and the inequalities are not strict (i.e. the inequalities signs are only $\leq, \geq, =$), then the feasibility problem becomes polynomial in the size of the system ([NW88]). Changing from natural variable to real variables in a system does cause an increment in the number of solutions, hence we may find a real solution without finding a natural one. With the aim to disprove such a property we define the following feasibility system.

Definition 2.3.2. *Let $G = (V, E)$ be a k-colored graph. We call* real balance system *for G the following system of equations on the set of variables $\{x_e \mid e \in E\}$.*

$$\begin{array}{lrl}
\textit{1. for all } v \in V & \sum_{e \in v^{\rightarrow}} x_e & = \sum_{e \in v^{\leftarrow}} x_e \\
\textit{2. for all } a \in [k-1] & \sum_{e \in E(a)} x_e & = \sum_{e \in E(k)} x_e \\
\textit{3. for all } e \in E & x_e & \geq 0 \\
\textit{4.} & \sum_{e \in E} x_e & \geq 1 \\
\textit{5. for all } e \in E & x_e & \in \mathbb{R}.
\end{array}$$

Let $m = |E|$ and $n = |V|$, the balance system has m variables and $m + n + k$ constraints.

Observe that not only we asked the variable to be in the set of reals number, but also the fourth constraint has became $\sum_{e \in E} x_e \geq 1$. This has been done with the aim to remove the constraint with the $>$ sign which would not ensure a polynomial resolvability of the feasibility problem. Also the substitution is reasonable because the fourth constraint was just added to avoid the solution with all zero components, this is also achieved by the new constraint. However, the new constrain seems to be more restrictive, nonetheless the fact that we are using real numbers make it flexible enough, as shown by the following lemma.

Lemma 2.3.3. *There exists a solution to the real balance feasibility problem if and only if there exists a solution to the natural balance feasibility problem.*

if. Suppose there exist a vector $x_e : E \to \mathbb{N}$ solution to the natural balance system. This solution satisfy all constraints of the real version of the feasibility problem but the fourth. Hence, we compute a new function $y_e : E \to \mathbb{R}$ such that for all edged $e \in E$ we have $y_e = \frac{x_e}{\sum_{e \in E} x_e}$. It is easy to see that it satisfies all constraints of the real balance system.

[only if] If the real balance system is feasible, since it has integer coefficients, it has to have a rational solution. Such rational solution $x_e : E \to \mathbb{Q}$, satisfies also all the constraints of the natural balance system but for the fact that its values do not belong to $\mathbb{N}$. Therefore, if d is the minimum common multiple among the denominators of the rational numbers x_e, we can define a new vector $y_e : E \to \mathbb{N}$ such that for all edges $e \in E$ it holds $y_e = d \cdot x_e$. Since, all constraints of the natural system are either equalities or inequalities of the type $a^T x \sim 0$, for $\sim \in \{>, \geq\}$, they are also satisfied by the function y_e.

□

The following is a corollary of Theorem 2.2.2, and Lemmas 2.3.2 and 2.3.3.

Corollary 2.3.1. *If G is strongly connected, there exists a balanced path in G if and only if the real balance system for G is feasible.*

At this point, computing a rational solution to the real balance system provides us with a set of loops representing a balanced path in a strongly connected graph. On the other hand, if we do not find such a solution we conclude that there are no loops with zero as difference value of an n.l.c. and, hence, no balanced path.

Theorem 2.3.1. *The balance problem, i.e. determining whether on a graph G there exists a balanced path and computing a representation of such path if it exists, is polynomial in the size of G.*

Proof. In order to solve the balance problem in G, first we compute the maximal connected components of G using the classical Tarjan's algorithm [CLRS01]. This algorithm is polynomial in n and m. Then, in each component we compute whether the real balance system is feasible, by using the polynomial algorithm for feasibility of sets defined by linear constraints [NW88]. This second algorithm is used at most n times and it is polynomial in the number of constraints $(n + m + k)$ and in the logarithm of the maximum modulus of a coefficient in a constraint (in our case, the maximum modulus is 1). If the feasibility algorithm answer positively to the existence of a solution, it also provide a rational one. By the proof of Lemma 2.3.2, such a solution allows us to compute in polynomial time a set of connected simple loops and the coefficients of an n.l.c. of value zero. As shown in the *if* part of the proof of Lemma 2.2.6, this in turn allows us to constructively characterize a balanced path in the graph.

□

2.4 Solving the Bounded Difference Problem

In the previous section we defined a feasibility system that allows us to compute a set of simple loops having zero as difference value of a natural linear combination. Such set of loops is also what is needed to determine a bounded path, provided that is is also overlapping. Hence, we need to add some more constraints that impose the loops found actually satisfy the said property. Lemma 1.2.2 allow us to link this requirement to a reachability problem, which is also commonly solved as flow problem in operative research. According the the lemma, in order to ensure overlapping of the loops, we need to find a node on such loops connected to all the other nodes only through edges belonging to the loops themselves. In the following x_e represents the variable associated to the edge e in the real balance system, and is called x-load associated to e.

In order to evaluate the connection between the nodes of the loops and a fixed node u, we imagine there is a connection flow generated from u and moving through the graph, only on edges belonging to the loops. Eventually some flow is absorbed by every node in the loops, thus ensuring that u is connected to all of them. In order to represent this flow, we introduce new variables y_e representing the amount of connection flow passing through the edges e. At this point, the constraints need to ensure that *(i)* there is some absorbed flow on all the nodes belonging to the loops, *(ii)* the absorbed flow is generate by the fixed node u, *(iii)* there is a connection flow only on edges belonging to the loops. The second problem is solved by observing that only nodes v belonging to the loops satisfy the property $\sum_{e \in v^{\leftarrow}} x_e > 0$ while for the other nodes, the entering x-load is zero. Hence, we can imagine that every node absorbs a connection load equal to $\sum_{e \in v^{\leftarrow}} x_e$ and be sure that all and only the nodes belonging to the loops will absorb some of it. We

impose such condition by asking that for each node v, the exiting y-load is less than the entering y-load by the quantity $\sum_{e \in v^{\leftarrow}} x_e$, i.e., we use the constraint $\sum_{e \in v^{\leftarrow}} y_e - \sum_{e \in v^{\rightarrow}} y_e = \sum_{e \in v^{\leftarrow}} x_e$. For the third point we just need to ask that u generates as much y-load as that absorbed by the other nodes, hence, $\sum_{e \in u^{\rightarrow}} x_e - \sum_{e \in v^{\leftarrow}} x_e = \sum_{v \in V \setminus \{u\}} \sum_{e \in v^{\leftarrow}} x_e$. For the third point, since edges are used only if $x_e > 0$, a constraint of the form $0 \leq y_e \leq C \cdot x_e$ for some constant C is enough. The value y_e of a flow on e is bounded by the total flow connecting u to the other nodes, i.e., by the value $\sum_{v \in V \setminus \{u\}} \sum_{e \in v^{\leftarrow}} x_e$. Observe that this flow is also bounded by $\sum_{v \in V} \sum e \in v^{\leftarrow} x_e$. Since all the edges enter in one and only one node the above value is equal to $\sum_{e \in E} x_e$. Hence, in order to evaluate the size of C we need to determine the maximum value of $\sum_{e \in E} x_e$ and the minimum value of x_e in a solution of the real balance system. This values exists and we determine them in the following lemma.

Lemma 2.4.1. *Let $G = (V, E)$ be a k-colored graph, with $|V| = n$, $|E| = m$, and $s_G = min\{n + k - 1, m\}$. For all solutions x to the real balance system for G there exists a solution x' such that, for all $e \in E$, it holds $(x_e = 0 \Rightarrow x'_e = 0)$ and $(x_e > 0 \Rightarrow 1 \leq x'_e \leq s_G!)$. As a consequence, $1 \leq \sum_{e \in E} x'_e \leq m \cdot s_G!$.*

Due to its many technical details, we postpone the proof of this lemma later in this section (but the interested reader can already read it). The lemma assure us that whenever there is a solution to the real balance system there is another solution such that all components x_e are no smaller than 1 and the sum $\sum_{e \in E} x_e$ is not greater than $m \cdot s_G!$. Hence, the constant C we are looking for in the third requirement for overlapping of loops, is equal to $C = m \cdot s_G!$.

Once, formalized all the needed requirements, we can define the complete linear system of inequalities that allow us to solve the bounded problem.

Definition 2.4.1. *Let $G = (V, E)$ be a k-colored graph with $m = |E|$, $n = |V|$, and $s_G = \min\{n + k - 1, m\}$, and let $u \in V$ be a node. We call* bounded difference system *for (G, u) the following system of equations on the set of variables $\{x_e, y_e \mid e \in E\}$.*

1-5. The same constraints as in the balance system for G		
6. for all $v \in V \setminus \{u\}$	$\sum_{e \in v^{\leftarrow}} y_e - \sum_{e \in v^{\rightarrow}} y_e$	$= \sum_{e \in v^{\rightarrow}} x_e$
7.	$\sum_{e \in u^{\rightarrow}} y_e - \sum_{e \in u^{\leftarrow}} y_e$	$= \sum_{v \in V \setminus \{u\}} \sum_{e \in v^{\rightarrow}} x_e$
8. for all $e \in E$	y_e	≥ 0
9. for all $e \in E$	y_e	$\leq (m \cdot s_G!) x_e$
10. for alle $\in E$	y_e	$\in \mathbb{R}$.

The bounded difference system has $2m$ variables and $3m + 2n + k$ constraints.

The following lemma states that the bounded difference system can be used to solve the bounded difference problem.

Lemma 2.4.2. *There exists an overlapping set of simple loops in G, passing through a node u and having zero as difference value of an n.l.c. if and only if the bounded difference system for (G, u) is feasible.*

only if. Let $\mathcal{L}$ be an overlapping set of simple loops having zero as difference value of an n.l.c. with coefficients c_σ for all loops $\sigma \in \mathcal{L}$. Then by Lemma 2.3.2 we can construct a solution $x_e = \sum_{\sigma \in \mathcal{L}} c_\sigma |\sigma|_e$ to the bounded-difference system. By Lemma 2.4.1, there exists another solution $x' \in \mathbb{R}^m$ to the balance system, such that $x_e = 0 \Rightarrow x'_e = 0$ and $x_e > 0 \Rightarrow 1 \leq x'_e \leq s_G!$. If any loop of the overlapping set $\mathcal{L}$ passes through u, by Lemma 1.2.2, there exists a path ρ_v from u to any node v occurring in $\mathcal{L}$. We set $y_e = \sum_{v \in V' \setminus \{u\}} (|\rho_v|_e \sum_{e \in_v E} x'_e)$. Simple calculations show that (x', y) is a solution to the bounded difference system for (G, u).

[if] If there exists a vector $(x, y) \in \mathbb{R}^{2m}$ satisfying the bounded-difference system, then like we did in the second part of Lemma 2.3.2, using x, we can construct a set of simple loops $\mathcal{L}$ having zero difference value of an n.l.c. Since $\sum_{e \in u^\rightarrow} y_e - \sum_{e \in u^\leftarrow} y_e = \sum_{v \in V \setminus \{u\}} \sum_{e \in v^\rightarrow} x_e$, we have that u belongs to at least one edge used in the construction of $\mathcal{L}$. If we set $G' = (V', E')$ as the subgraph of G induced by $\mathcal{L}$, we are able to show by contradiction that there is a path in G' from u to every other node of V'. Indeed if for some $v \in V' \setminus \{u\}$ there is no path in G' from u to v then there is some load exiting from u that cannot reach its destination using only edges of G'. Since the constraints (8) make it impossible to carry load on edges of G that are not used in $\mathcal{L}$, the connection constraints cannot be satisfied. So, for all $v \in V'$ there is a path in G' from u to v. By Lemma 1.2.2, $\mathcal{L}$ is overlapping. □ □

At this point, computing a rational x-solution to the real balance system provides us with a set of overlapping loops representing a bounded path in a strongly connected graph. On the other hand, if we do not find such a solution we conclude that there are no loops with zero as n.l.c. and, hence, no balanced path.

Theorem 2.4.1. *The bounded problem, i.e. determining whether on a graph G there exists a bounded path and computing a representation of such path if it exists, is polynomial in the size of G.*

Proof. In order to solve the bounded difference problem in G, for all $u \in V$ we check whether the bounded difference system for (G, u) is feasible, by using a polynomial time algorithm for feasibility of linear systems [NW88]. This algorithm is used at most n times and it is polynomial in the number of constraints $(2n + 3m + k)$ and in the logarithm of the maximum modulus M of a coefficient in a constraint. In our case, $M = m \cdot s_G!$. Using Stirling's approximation, we have $\log(m \cdot s_G!) = \log(m) + \Theta(s_G \log(s_G))$. Therefore, we obtain the following.

If the feasibility algorithm answer positively to the existence of a solution, it also provide a rational one. By the proof of Lemma 2.3.2, such the x-component of the solution allows us to compute in polynomial time a set of simple loops and the coefficients of an n.l.c. of difference value zero. By the properties ensured by the y-component of the solution we know that the loops are overlapping. As shown in the proof of Lemma 2.2.6, this in turn allows us to constructively characterize a balanced path in the graph. □

The rest of this section is dedicated to the proof of Lemma 2.4.1.

We first introduce two preliminary lemmas.

Lemma 2.4.3. *Let $t \in \mathbb{N}$ be a natural number and $A \in [t]_0^{m \times m}$ be a square matrix, then $|det(A)| \leq t^m m!$. Moreover, if A is not singular then $|det(A)| \geq 1$.*

Proof. We prove the first statement by induction on m.

1. If $m = 1$ then $|det(A)| = |a_{1,1}| \leq t$.

2. If the statement holds for $m-1$, then for any $j \in [m]$ it holds that

$$det(A) = \sum_{i=1}^{m} (-1)^{i+j} a_{i,j} det(M_{i,j}),$$

where $M_{i,j} \in [t]_0^{(m-1)\times(m-1)}$ is a matrix obtained from A by removing the i-th row and the j-th column. So, $|det(A)| \leq |\sum_{i=1}^{m} a_{i,j} det(M_{i,j})| \leq \sum_{i=1}^{m} |a_{i,j}||det(M_{i,j})| \leq \sum_{i=1}^{m} t \cdot t^{m-1}(m-1)! = (tm)t^{m-1}(m-1)! = t^m m!$.

Moreover, if A is not singular, since A has an integer determinant it must be $|det(A)| \geq 1$. □

Lemma 2.4.4. *Let t be a natural number and $A \in [t]_0^{n\times m}$, $A' \in [t]_0^{n'\times m}$, $B \in [t]_0^{n\times 1}$, and $B' \in [t]_0^{n'\times 1}$ be four matrices. Let $S = \{x \in \mathbb{R}^m \mid Ax \geq B, A'x \geq B', x \geq \mathbf{0}\}$ and $M = min\{n + n', n + m\}$. If S is not empty, then there exists a vector $x \in S$ such that $x \in \mathbb{Q}^m$ and every component x_i is less than or equal to $k = M!t^M$.*

Proof. Let $I \in \mathbb{N}^{n\times n}$ be the identity matrix. First, we convert every inequality of the system $Ax \geq B$ in an equivalent equality by adding a new variable: the inequality $\sum_{i=1}^{m} a_{i,j}x_i \geq b_j$ becomes $\sum_{i=1}^{m} a_{i,j}x_i = b_i + y_j$ with $y_j \geq 0$. If we set $C = \left(\begin{smallmatrix} A & A' \\ -I & \mathbf{0} \end{smallmatrix}\right) \in \mathbb{R}^{(n+n')\times(n+m)}$, and $B'' = \left(\begin{smallmatrix} B \\ B' \end{smallmatrix}\right)$, we can define the set $S' = \{(x, y) \in \mathbb{R}^{m+n} \mid C\left(\begin{smallmatrix} x^T \\ y^T \end{smallmatrix}\right) = B'', (x, y) \geq \mathbf{0}\}$. It is easy to see that $S = \{x \in \mathbb{R}^m \mid \exists y \in \mathbb{R}^n . (x, y) \in S'\}$, thus in our hypothesis S' is not empty. We denote as r the rank of C, so $m \leq r \leq M$ since $-I$ is not singular. By a well known result in linear programming (see, for instance, Theorem 3.5 of [NW88]), the set S' contains a *basic* solution, i.e. there exists a non-singular submatrix $C' \in \mathbb{R}^{r\times r}$ of C, given by the columns $i_1, \ldots, i_r$ and by the rows $j_1, \ldots, j_r$ of C, such that in S there is the point $(z_1, \ldots, z_{m+n}) \in \mathbb{R}^{m+n}$ such that $z' = (z_{i_1}, \ldots, z_{i_r})$ is the unique solution to the system of linear equations $Cz'^T = (b''_{j_1}, \ldots, b''_{j_r})^T = B'''$, and for all $j \notin \{i_1, \ldots, i_r\}$ it holds that $z_j = 0$. By Cramer's theorem, for all $k \in [r]$ we have $z_{i_k} = det(C'_{i_k})/det(C')$ where C'_{i_k} is the matrix obtained from C' by replacing the i_k-th column with the matrix B'''. So, z' and z have components in $\mathbb{Q}$. Since $C', C'_{i_1}, \ldots, C'_{i_r} \in [t]_0^{r\times r}$, by Lemma 2.4.3 $|det(C_i)| \leq r!t^r$. Moreover, since C' is not singular we have $|det(C)| \geq 1$. In conclusion, $z_{i_k} \leq |det(C_i)|/|det(C)| \leq (r)!t^r \leq M!t^M$, as requested. □

Now, we are ready to prove Lemma 2.4.1.

Lemma 2.4.5. *Let $G = (V, E)$ be a k-colored graph, with $|V| = n$, $|E| = m$, and $s_G = min\{n + k - 1, m\}$. For all solutions x to the balance system for G there exists a solution x' such that, for all $e \in E$, it holds $(x_e = 0 \Rightarrow x'_e = 0)$ and $(x_e > 0 \Rightarrow 1 \leq x'_e \leq s_G!)$. As a consequence, $1 \leq \sum_{e\in E} x'_e \leq m \cdot s_G!$.*

Proof. Let x be a solution to the balance system for G, and let J be the set of all edges e such that $x_e > 0$. By construction, $|J| > 0$. We represent the first two sets of equalities of the balance system in matrix form as $Dx = \mathbf{0}$. Then, the set of points satisfying the balance system is $P = \{y \in \mathbb{R}^m \mid Dy = \mathbf{0}, y \geq \mathbf{0}, \sum_{e \in E} y_e > 0\}$. Now the subset of P, $P' = \{y \in P \mid \forall e \in J \,.\, y_e \geq 1 \text{ and } \forall e \notin J \,.\, y_e = 0\} = \{y \in P \mid \forall e \in E \,.\, (x_e > 0 \Rightarrow y_e > 1) \text{ and } (x_e = 0 \Rightarrow y_e = 0)\}$ is not empty. Indeed, the vector $z = x(\min_{e \in J} x_e)^{-1}$ is in P', since (i) $Dz = (\min_{e \in J} x_e)^{-1} Dx = \mathbf{0}$, (ii) for all $e \in J$, we have $z_e = x_e(\min_{e \in J} x_e)^{-1} \geq 1$, and (iii) for all $e \notin J$, we have $z_e = 0$.

The set of inequalities "$\forall e \in J \,.\, y_e \geq 1$" can be represented as the system of linear equations $Fy \geq \mathbf{1}$, with $\mathbf{1} \in \{1\}^{l \times 1}$. Similarly, the set of equalities "$\forall e \notin J \,.\, y_e = 0$" can be represented as $F'y = \mathbf{0}$. If we define $D' = \binom{D}{F'} \in \{-1, 0, 1\}^{(2n+k-l-1) \times m}$, we have $P' = \{y \in \mathbb{R}^m \mid D'y = \mathbf{0}, Fy \geq \mathbf{1}\}$. Since $D', F, \mathbf{1}, \mathbf{0}$ all have elements in $\{-1, 0, 1\}$, by Lemma 2.4.4 the set P contains an element $x' \in \mathbb{Q}^m$ such that for all $i \in [m]$, $x'_i \leq (\min\{2n + k - 1, l + m\})! \leq (\min\{2n + k - 1, n + m\})! = s_G!$, which concludes the proof. □

2.5 The perfectly balanced finite path problem

In this section, we introduce an NP-hard problem similar to the bounded difference problem. Given a k-colored graph G and two nodes u and v, the new problem asks whether there exists a perfectly balanced path from u to v. We call this question the *perfectly balanced finite path problem*. To see that this problem is closely related to the bounded difference problem, one can note that it corresponds to the statement of item 3 in Lemma 2.2.1, by changing the word *loop* to *finite path*. In the following we prove that such a problem is NP-complete.

We ffirst prove that the problem is NP-hard by by a reduction from 3SAT which is known to be NP-hard [CLRS01].

Theorem 2.5.1. *The perfectly balanced finite path problem is NP-hard.*

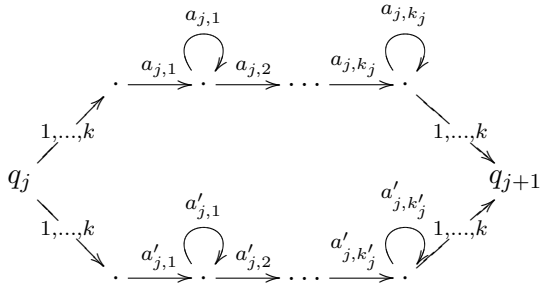


Figure 2.2: Proof of Theorem 2.5.1: The j-th subgraph G_j of G.

Proof. We prove the statement by a reduction from 3SAT which is known to be an NP-hard problem [CLRS01].

Given a 3SAT formula φ on n variables $x_1, \ldots, x_n$ with k clauses $C_1, \ldots, C_k$, we construct a k-colored graph G such that each color i is associated with the clause C_i. Precisely, for each

variable x_j, we construct a subgraph G_j of G with a starting node q_j and an ending node q_{j+1}, as shown in Figure 2.2. For $1 \leq j \leq n$, the labels $a_{j,1}, \ldots, a_{j,k_j}$ are the colors corresponding to the clauses in which x_j occurs affirmed and $a'_{j,1}, \ldots, a'_{j,k'_j}$ are the colors of the clauses in which x_j occurs negated. Moreover, the edges labeled with $1, \ldots, k$ concisely represent a sequence of k edges, each labeled with a different color. Finally, the graph G is obtained by concatenating each graph G_j with G_{j+1}, as they share the node q_{j+1}, for $1 \leq j < n$.

We show that the formula φ is satisfiable if and only if there exists a perfectly balanced path in G from q_1 to q_{n+1}.

First, assume that φ is satisfiable. Then, there exists a truth assignment for the variables that satisfies φ. Using this assignment, we construct a perfectly balanced path in which each color appears exactly $2n + 3$ times. In particular, for all subgraphs G_j, the path takes the upper branch if x_j is assigned true and the lower branch otherwise. For each clause C_i, let L_i be the indexes of the variables that render C_i true, under the given truth assignment. We obtain that the constructed path passes through at least $2n + |L_i|$ non-self-loop edges colored with i. This holds because at each subgraph it passes through the edges labeled $1, \ldots, k$ once at the beginning and once at the end. Moreover, for all $j \in L_i$, the path passes through another non-self-loop edge labeled with i in G_j. Since $|L_i| \geq 1$, the path may pass through a self-loop labeled with i at least once in the graph. Thus, by taking $3 - |L_i|$ times one of those self-loops, we get the desired number $2n + 3$ of occurrences of i, for all colors i.

Conversely, assume that there exists a perfectly balanced path from q_0 to q_{n+1}. For all subgraphs G_j the path takes either the upper or the lower branch. Then, there are two possible situations:

1. Each color occurs $2n + l$ times with $l \geq 1$. We define the assignment in the following way: we set x_j to *true* if the path takes the upper branch in the subgraph G_j, and to *false* otherwise. We claim that such assignment satisfies φ. For all colors i the path passes through an i-colored edge α such that it is not a self-loop and it is not a starting or an ending edge of a subgraph G_j (those edges are the first $2n$). Such edge α is on a branch of a subgraph G_j, consequently the assignment for x_j satisfies the clause C_i. Being i arbitrary, all clauses C_i are satisfied by the assignment of the variable.

2. Each color occurs $2n$ times in the path. We define the assignment as follows: we set x_j to *true* if the path takes the lower branch in G_j, and to *false* otherwise. We claim that such assignment satisfies φ. For all colors i there exists at least one variable x_j appearing in the clause C_i. However, the path does not pass through any edge colored with i, except the mandatory edges at the beginning and end of each G_j. Then, in G_j the path takes the branch opposite to the assignment of x_j that makes C_i true Then, the opposite assignment of x_j (the one we choose) makes C_i true.

□

We prove that the problem is in NP by reducing it to the feasibility of a system of linear equation with integer variables, which is known to be NP-complete [NW88]. This time the inequality signs in $\{<, >\}$ are allowed. Before describing the related linear system, we recall a result of integer programming presented in [Sch86].

Lemma 2.5.1 ([Sch86]). *Let $A \in \mathbb{Z}^{n\times n}$ and $B \in \mathbb{Z}^{n\times 1}$. Let $S = \{x \in \mathbb{Z}^n \mid Ax \leq B\}$ be an integer convex set. If S is not empty then there exists a point $x \in S$ such that the sum of the components of x is bounded by $6n^3\varphi$, where φ is the maximum sum of the coefficients of an inequality of the system $Ax \leq B$.*

Definition 2.5.1. *Let $G = (V, E)$ be a k-colored graph and $u, w \in V$ be two distinct nodes. We call* perfectly balanced path system *for (G, u, w) the following system of equations on the set of variables $\{x_e, y_e \mid e \in E\}$.*

$$
\begin{array}{lrl}
\textit{1. for all } v \in V \setminus \{u, w\} & \sum_{e\in v^{\leftarrow}} x_e & = \sum_{e\in v^{\rightarrow}} x_e \\
\textit{2.} & \sum_{e\in u^{\leftarrow}} x_e & = 1 + \sum_{e\in u^{\rightarrow}} x_e \\
\textit{3.} & \sum_{e\in w^{\leftarrow}} x_e & = -1 + \sum_{e\in w^{\rightarrow}} x_e \\
\textit{4. for all } a \in [k-1] & \sum_{e\in E(a)} x_e & = \sum_{e\in E(k)} x_e \\
\textit{5. for all } e \in E & x_e & \geq 0 \\
\textit{6.} & \sum_{e\in E} x_e & > 0 \\
\textit{7. for all } v \in V \setminus \{u\} & \sum_{e\in v^{\leftarrow}} y_e - \sum_{e\in v^{\rightarrow}} y_e & = \sum_{e\in v^{\rightarrow}} x_e \\
\textit{8.} & \sum_{e\in u^{\rightarrow}} y_e - \sum_{e\in u^{\leftarrow}} y_e & = \sum_{v\in V\setminus\{u\}} \sum_{e\in v^{\rightarrow}} x_e \\
\textit{9. for all } e \in E & y_e & \geq 0 \\
\textit{10. for all } e \in E & y_e & \leq (6(d-1)^3\varphi)x_e. \\
\textit{11. for all } e \in E & x_e, y_e & \in \mathbb{Z}
\end{array}
$$

where φ is the maximum sum of the coefficients of an inequality in the first six sets of constraints. Let $m = |E|$ and $n = |V|$, the perfectly balanced path system has $2m$ variables and $3m + 2n + k$ constraints.

It helps to think of the vectors x and y as two integer loads associated to the edges of G. The constraints $1 - 6$ are almost the same constraints of the natural balance problem for G, and they ask that x should represent a path from u to w and a set of simple loops such that the latter have a n.l.c. equal to the opposite of the color difference vector of the path.

The constraints $7 - 10$ are *connection constraints*, asking that y should represent a connection load, from u to every other node of the simple loops defined by x, and carried only on the edges of those loops. Thus, the constraints $7 - 10$ ask that the loops represented by x should be reachable from u, using only edges represented by x, similarly to the bounded difference system of Section 2.4. The only difference is the bound in the constraints 10, which is justified by Lemma 2.5.1.

Lemma 2.5.2. *There exists a perfectly balanced path in G from u to w if and only if the perfectly balanced path system (G, u, w) is feasible.*

The proof of the previous lemma is similar to that for the balance problem. Since the feasibility problem for an integer linear system is in NP, and by Theorem 2.5.1, we obtain the following.

Theorem 2.5.2. *The perfectly balanced finite path problem is NP-complete.*

...

...

3 Priority Scheduling

Contents

3.1 Frequency Goals

In the previous chapter we defined fairness requirements without actually quantifying the resources dedicated to an activity. Here, we try to measure the amount of such resources through frequency.The frequency of an activity can be expressed as the relative number of its occurrences with respect to the overall number of activities performed.

Definition 3.1.1. Frequency

1. *The frequency of a color i on a finite sequence $x \in [i]^*$ is the value $\frac{|x|_i}{|x|}$.*
2. *The frequency of a color i on a finite path in a k-colored graph is the frequency of color i on the associated color sequence.*
3. *The frequency vector of a finite sequence $x \in [i]^*$ is the vector $(\frac{|x|_1}{|x|}, \dots, \frac{|x|_k}{|x|}) \in \mathbb{Q}^k$*
4. *The frequency vector of a finite path in a k-colored graph is the frequency vector of the associated color sequence.*

Since a system's run is infinite, we can only evaluate such a frequency at every finite prefix. Thus, we can assume that the frequency of an infinite run is the asymptotic value to which the frequencies along the prefixes tend to.

Definition 3.1.2. Asymptotic frequency

Let $x \in [k]^\omega$ be an infinite sequence of colors. The asymptotic frequency f_i of color $i \in [k]$ on x is the limit (if it exists) of the frequencies of i on all finite prefixes of x, i.e., $f_i = \lim_{m \to +\infty} \frac{|x^{\leq n}|_i}{n}$. The asymptotic frequency of x is the real vector $f \in [0,1]^k$ such that each component f_i is the asymptotic frequency of i on x. A path ρ on a k-colored graph has asymptotic frequency $f \in [0,1]^k$ (resp. admits frequency $f_i \in [0,1]$ for color i) if its color sequence does.

Example 3.1.1. *The color sequence $x = \prod_{i=1}^{+\infty}((110)^i 11) \in [1]^\omega$ has asymptotic frequency vector $f = (\frac{1}{3}, \frac{2}{3})$.*

An asymptotic frequency has a drawback: it disregards all those contributions of execution of activities that grows sublinearly in the length of the prefix. In the previous example, we can observe that activity 1 is executed more than $\frac{2}{3}$ of the times, but the difference between the real frequencies of activity grows so slowly that it is asymptotically negligible. Indeed, by exploding the definition of limit for the asymptotic frequency, it is possible to find the following equivalent formulation.

Lemma 3.1.1. *An infinite color sequence $x \in [0,1]^\omega$ has asymptotic frequency $f \in [0,1]^k$ if and only if there exist a function $g : N \to N$ such that $g(n) \in o(n)$ and for every finite prefix $x^{\leq n}$ the number of occurrences of color i on the prefix does not differ from the expected $f_i \cdot n$ more than $g(n)$, i.e., it belongs to the set $[f_i \cdot n - g(n), f_i \cdot n + g(n)]$.*

So, an asymptotic frequency allows intervals in the run that are not compliant with the frequency itself, provided that their uncompensated contributions is small enough, and, hence, negligible at infinite. However, the smaller the increment of the function $g(n)$, the smaller this undesired contribution grows. The smallest growing function $g(n)$ is the constant function, which completely halts the growth of this frequency non-compliant contribution, indeed when $g(n) = C$ for each color we can expect that the number of occurrences on a prefix of length n is very close $n \cdot f_i$ but for that constant. When $g(n)$ is constant, we call the asymptotic frequency uniform to mean that the unwanted contributions eventually stops or are always compensated. The smaller is the constant the sooner the non-compliant contributions are compensated by a following one.

Definition 3.1.3. Uniform (asymptotic) frequency

Let $x \in [k]^\omega$ be an infinite sequence of colors, with asymptotic frequency $f \in [0,1]^k$. f is said uniform on x if there exists a constant C such that for every finite prefix $x^{\leq n}$ of length n, and every color $i \in [k]$, the frequency of color i oscillates in the set $[f_i - \frac{C}{n}, f_i + \frac{C}{n}]$.

Example 3.1.2. *The color sequence $x = \prod_{i=1}^{+\infty}((110)^i 11) \in [1]^\omega$ has asymptotic non-uniform frequency vector $f = (\frac{1}{3}, \frac{2}{3})$. The color sequence $y = (110)^\omega$ has uniform frequency vector $f = (\frac{1}{3}, \frac{2}{3})$.*

It is possible to evaluate different degree of uniformity by choosing a sublinear function $g(n)$ in the equivalent definition of asymptotic frequency given by Lemma 3.1.1. However, we leave such a discussion to Section 3.6 By assigning to each activity a different frequency depending on its priority, provided that all frequency sum to 1, we are able to evaluate whether a path satisfies the given priority policy.

Definition 3.1.4. *Let $f \in [0,1]^k$ be a vector of real number such that $\sum_{i=1}^k f_i = 1$. Then, the asymptotic (resp. uniform) f-frequency goal W_{af} (resp. W_{uf})is the set of all infinite color sequences on $[k]$ with asymptotic (resp. uniform)frequency f.*

In the following, we call $f \in \mathbb{R}^k$ a frequency vector only if $f \in [0,1]^k$ and $\sum_{i=1}^k f_i = 1$.

3.2 Graphs Characterization

Like we did in Section2.2, in this section, we characterize the existence of an asymptotic frequency-f or uniform frequency-f paths by means of properties of the underlaying colored graph. In particular, we observe that we can compose simple loops for the construction of a priority path. On the converse, we prove that a priority paths always contain some regularities, that allow us to extract loop components. By evaluating the properties of such loops, we show that whenever there exist of a priority path, a set of simple loops can be used to construct some regular priority path.

3.2.1 Frequency-f problem

The asymptotic frequency-f property states that, along the prefixes of a path, the ratio between the number of occurrences of a color $i \in [k]$ and the length of the prefix, does not differ from f_i, but for a value which grows asymptotically slower than any function polynomial in the length of

the prefix. Since, such a ratio has a limited interval of variability, like the color difference in the balanced problem, is it maybe possible that even in this case there is an unavoidable periodicity? Observe that if we want to construct a periodic path $\rho = \sigma^\omega$ with asymptotic frequency f, then is is sufficient that the loop σ is endowed with such a frequency, i.e., that for all color $i \in [k]$ it holds that $\frac{|\sigma|_i}{|\sigma|} = f_i$. If we decompose σ in a combination of a sequence of loops $\mathcal{L} = \sigma_1, \ldots, \sigma_l$, then we obtain that for each $i \in [k]$ it holds that $f_i = \frac{\sum_{j=1}^{l} |\sigma_j|_i}{\sum_{j=1}^{l} |\sigma_j|_i}$. Hence, the frequency is like an average of vectors $\overline{N_{um}}\sigma_i$ with weights $|\sigma_i|$.

Definition 3.2.1. Ratio of an n.l.c

In a k-colored graph, the ratio of natural linear combination $\mathcal{T} = \{(\sigma_1, c_1) \ldots (\sigma_l, c_l)\}$ is the vector $\frac{\sum_{i=1}^{l} c_i \cdot \overline{N_{um}}\sigma_i}{\sum_{i=1}^{l} c_i \cdot |\sigma_i|}$.

In general, we capture the ability of weighted natural vectors to form a weighted average through the following definition.

Definition 3.2.2. Weighted Ratio

Let $A = \{(A_1, w_1), \ldots, (A_m, w_m)\} \subseteq \mathbb{Z}^d \times \mathbb{N}_{-0}$ be a finite set of m pairs (integer vector, respective weight), we call combination value of A with respect to coefficients $c_1, \ldots, c_n \in \mathbb{N}$ (such that they are not all zeros) a vector $D = \sum_{i=1}^{m} c_i A_i$, Moreover, we define the weight of A as $n_D = \sum_{i=1}^{m} c_i w_i$ and the ratio of A as $\frac{D}{n_D}$.

At this point the question about the existence of a periodicity in a path with asymptotic frequency f, relates to the ability of the occurrence vectors of loops involved in the path, to form a ratio of value f. The next lemma shows that an infinite weighted sum of weighted natural vectors cannot converge to a real vector f, unless the set of vectors used can form a ratio with value f.

Lemma 3.2.1. *Let $f \in \mathbb{Q}^d$, and $A \subset \mathbb{Z}^d \times \mathbb{N}$ be a finite set such that no n.l.c. of A has ratio f. Let $\{(B_n, u_n)\}_n$ be an infinite sequence of elements of A, $S_n = \sum_{l=0}^{n} B_l$ be the partial sum, and $U_n = \sum_{l=0}^{n} u_l$ be the partial sum of the weights. Then, there exist an index $i \in [d]$ such that $\lim_{n \to +\infty} \frac{S_{n,i}}{U_n} \neq f_i$.*

Proof. Let $A = \{(A_1, w_1), \ldots, (A_m, w_m)\}$ and $g : \mathbb{R}^m \mapsto \mathbb{R}_+$ be the function such that $g(c_1, \ldots, c_m) = \max_{1 \leq i \leq d} \left\{ \left| \frac{\sum_{n=1}^{m} c_n A_{n,i}}{\sum_{n=1}^{m} c_n w_n} - f_i \right| \right\}$. First, note that g is a continuous function, since it is the maximum of continuous functions. Let now $K \subset \mathbb{R}^m$ be the set $\{(c_1, \ldots, c_m) \in [0,1]^m \mid \sum_{i=1}^{m} c_i = 1\}$. Note that $\mathbf{0} \notin K$ and that K is compact, since it is a finite dimensional space defined by a linear equation. Hence, by Weierstrass theorem, g admits a minimum $M = \min_{x \in K}\{g(x)\}$ on K. Since, by hypothesis, there is no n.l.c. of A with ratio f, M must be strictly positive. Indeed, if by contradiction $M = 0$, there should be a non-zero vector $(c_1, \ldots, c_m) \in K$ such that for all $i \in [d]$,

$$\sum_{n=1}^{m} c_n A_{n,i} - L_i \sum_{n=1}^{m} c_n w_n = M = 0. \tag{3.1}$$

Since (3.1) is a homogeneous linear equation with rational coefficients and since it has a non-negative solution, by Lemma 2.2.4 it also has a non-negative *integer* solution with at least one

positive component. This solution induces a n.l.c. of A with ratio ff, contradicting the hypothesis on A.

Now, consider the sequence $\{(B_n, u_n)\}_n$, its partial sums $S_n = \sum_{l=0}^{n} B_l$, and its weight partial sum $U_n = \sum_{l=0}^{n} u_l$. Moreover, let $\delta_{l,n}$ be the number of times for which the pair (A_l, w_l) occurs in the sequence up to position n and let $c_{l,n} = \delta_{l,n}/n$. Then $S_n = \sum_{l=1}^{m} \delta_{l,n} \cdot A_l = n \cdot \sum_{l=1}^{m} c_{l,n} \cdot A_l$ and $U_n = \sum_{l=1}^{m} \delta_{l,n} \cdot w_l = n \cdot \sum_{l=1}^{m} c_{l,n} \cdot w_l$. Since we have $\sum_{l=1}^{m} \delta_{l,n} = n$ for all $n \in \mathbb{N}$, it is obvious that $(c_{1,n}, \ldots, c_{m,n}) \in K$.

Let now $Z_n \in \mathbb{R}^{d \times d}$ be the matrix defined by $Z_{n,i} = \left| \frac{\sum_{l=1}^{m} c_{l,n} A_{l,i}}{\sum_{l=1}^{m} c_{l,n} w_l} - f_i \right|$. Since there is no n.l.c. of A with ratio f, it holds that for all $n \in \mathbb{N}$ there exists a non-zero element in Z_n. Let $\{i_n\}_n$ be an index sequence such that $Z_{n,i_n} = \max_{1 \leq i \leq d}\{Z_{n,i}\} > 0$. Since $\{i_n\}_n$ can assume at most d different values, there exists a pair i^* that occurs infinitely often in $\{i_n\}_n$. Let $\{h_t\}_t$ be the index sequence such that $i_{h_t} = i^*$ and there is no $t' \in]h_t, h_{t+1}[$ with $i_{t'} = i^*$. Then, consider the subsequence $\{Z_{h_t,i^*}\}_t$ of $\{Z_{n,i^*}\}_n$, Hence, $\lim_{t \to +\infty} Z_{h_t,i^*} \geq M > 0$ and consequently $\lim_{n \to +\infty} Z_{n,i^*} \neq 0$, whenever these limits exist. In conclusion, $\lim_{n \to +\infty} \frac{\sum_{l=1}^{m} c_{l,n} A_{l,i}}{\sum_{l=1}^{m} c_{c,l} w_l} = \lim_{n \to +\infty} \frac{S_{n,i}}{U_n} \neq f_i$. □

Whenever we have an path with color frequency f, we are able to decompose this path in its quasi-segmentation. Each loop is associated with a difference vector and a length, and every time a loop ends, the frequency vector of the prefix is the ratio between the sum of the occurrence vectors of the loops met so far and the sum of their length. Hence, if the loops cannot form a linear composition with frequency f, the path cannot have asymptotic frequency f.

Lemma 3.2.2. *Let G be a k-colored graph and ρ be an infinite path in G with color frequency $f \in \mathbb{R}^k$, then there exists a connected set of simple loops having an n.l.c. of ratio f.*

Proof. If there exists an infinite path ρ with asymptotic frequency f, since the set of nodes is finite, there is a set V' of nodes occurring infinitely often in ρ. Let ρ' be a suffix of ρ containing only nodes in V'. The path ρ' has frequency f and it is composed by an infinite sequence of simple loops on V' (see Definition 1.2.6 for further details). Let $\mathcal{L}$ be the (finite) set of such simple loops, and let $A = \{(\sigma, |\sigma|) \mid \sigma \in L)\}$ be the weighted set of occurrence vectors of the loops in $\mathcal{L}$. Every time a loop trough V' closes along ρ', the difference vector up to that point is the sum of the difference vectors of the simple loops occurred so far, plus the difference vector of the remaining simple path. Since the remaining simple path cannot have length greater than $|V'|$, the difference vector up to that point differs from a sum of a sequence of elements of A by a constant-bounded term. Let $n(j)$ be the index of the j-th point where a loop is closed along ρ'. Since ρ' has frequency f, each component of the difference sequence $\{\frac{\overline{N_{um}}(\rho'^{\leq n(j)})}{|\rho'^{\leq n(j)}|}\}_{i \in \mathbb{N}}$ tends to f_i. Let $\{\sigma_h\}_{h \in \mathbb{N}}$ be the sequence of loops such that $\{\sigma_h\}_{h=1}^{j}$ is the quasi-segmentation of $\rho'^{\leq n(j)}$ with rest r_j. Then we have $\overline{N_{um}}\rho'^{\leq n(j)} = (\sum_{h=1}^{j} \overline{N_{um}}(\sigma_h)) + \overline{N_{um}}(r_j)$ and $|\rho'^{\leq n(j)}|(\sum_{h=1}^{j} |\sigma_h|) + |r_j|$. Since r_j has at most length $|V|$ we have for all $i \in [k]$ the sequence $\{\frac{\sum_{h=1}^{j} |\sigma_j|_i}{\sum_{h=1}^{j} |\sigma_j|}\}_j$ converges to the same limit of the sequence $\{\frac{\sum_{h=1}^{j} |\sigma_j|_i + |r_j|_i}{\sum_{h=1}^{j} |\sigma_j|} + |r_j|\}_j$, i.e. f_i. By Lemma 3.2.1, A has an composition with ratio f. Then, the simple loops of $\mathcal{L}$, which occur with a positive coefficient in the composition of A with ratio f, are connected, because they are extracted from the same path π. □

Following a similar construction developed in Lemma 2.2.6, given a set of simple loops with a natural linear ratio f, we are able to construct a path with asymptotic frequency f. The path is obtained by using periodically all the loops as many times as requested by the natural linear ratio and by connecting them with connection paths used less and less often in order to make their contribution negligible at infinite. The following lemma formalizes the construction.

Lemma 3.2.3. *If a k-colored graph G contains a set of connected simple loops having an n.l.c. of ratio f, then there exists in G an infinite path ρ with frequency f.*

Proof. Let $\mathcal{L} = \{\alpha_0, \alpha_1, \ldots, \alpha_{h-1}\}$, and denote by v_i the first node of α_i in its representation as a cyclic sequence of nodes. For all $i = 0, \ldots, h-1$, let π_i a (possibly empty) path that starts in the last node of α_i and ends in the first node of $\alpha_{(i+1) \bmod h}$. Since $\mathcal{L}$ is connected, it is possible to find such paths. Let A_i be the color vector of α_i, and let A_i be the color difference matrix of π_i. Moreover, let $(c_0, c_1, \ldots, c_{n-1})$ be the non-negative integers such that $\frac{\sum_{i=0}^{h-1} c_i A_i}{\sum_{i=0}^{h-1} c_i n_i} = f$. Then, we define the vector $Z = \sum_{i=0}^{h-1} c_i A_i$. Finally, let n_i the number of edges in α_i and m_i the number of edges in π_i. At this point, we define $n = \sum_{i=0}^{h-1} c_i \cdot n_i$ and $m = \sum_{i=0}^{h-1} m_i$.

In order to construct a path with color limit L, we reason as follows. Since in general the loops in $\mathcal{L}$ do not share a node with each other, to move from α_i to α_{i+1}, we have to pay a price, represented by the color vector of π_i. In order to make this price disappear in the long-run, we traverse the loops α_i an increasing number of times: in the first round, we traverse it c_i times, in the second round, $2c_i$ times, and so on. Formally, the construction is iterative and at every round $i > 0$ we add, to the already constructed path, the cycle ρ_i defined by

$$\rho_i = \alpha_0^{ic_0} \pi_0 \alpha_1^{ic_1} \pi_1 \ldots \alpha_{h-1}^{ic_{h-1}} \pi_{h-1}.$$

Note that the cycle ρ_i starts and ends at node v_0 and contains $m + i \cdot n$ edges. The required infinite path is then $\rho = \rho_1 \rho_2 \ldots \rho_i \ldots$. We now show that this path has frequency f.

Let $l_0 = 0$ and, for all $i > 0$, let $l_i = \sum_{j=1}^{i} |\rho_j| = \sum_{j=1}^{i} (m + i \cdot n) = i \cdot m + \frac{i \cdot (i+1)}{2} n = \frac{i^2}{2} n + O(i)$, so that $\rho^{\leq l_i} = \rho_1 \ldots \rho_i$. We compute for all colors a the value $|\rho^{\leq l_i}|_a = \sum_{j=1}^{i} |\rho_j|_a = \sum_{j=1}^{i} (\sum_{q=0}^{h-1} |\pi_q|_a) + i(\sum_{q=0}^{h-1} c_q |\alpha_q|_a) = \frac{i^2}{2} \cdot (\sum_{q=0}^{h-1} c_q |\alpha_q|_a) + O(i)$. Consider an index $j \in \mathbb{N}$, there exists an index $i(j)$ such $l_{i(j)} \leq j \leq l_{i(j)+1}$. Since for all colors a, the functions $|\rho^{\leq j}|_a$ and $|\rho^{\leq j}|$ are increasing in j, we have that:

$$\frac{|\rho^{\leq l_{i(j)}}|_a}{|\rho^{\leq l_{i(j)+1}}|} \leq \frac{|\rho^{\leq j}|_a}{|\rho^{\leq j}|} \leq \frac{|\rho^{\leq l_{i(j)+1}}|_a}{|\rho^{\leq l_{i(j)}}|}$$

Since $\lim_{j \to +\infty} \frac{|\rho^{\leq l_{i(j)+1}}|_a}{|\rho^{\leq l_{i(j)}}|} = \frac{(i+1)^2 \cdot (\sum_{q=0}^{h-1} c_q |\alpha_q|_a)}{i^2 \cdot n} = f_a$, and $\lim_{j \to +\infty} \frac{|\rho^{\leq l_{i(j)}}|_a}{|\rho^{\leq l_{i(j)+1}}|} = f_a$, we have that the value of $\lim_{j \to +\infty} \frac{|\rho^{\leq j}|_a}{|\rho^{\leq j}|}$ is f_a. □

As an immediate corollary of the two previous lemmas, the next theorem follows.

Theorem 3.2.1. *Let G be a k-colored graph, there exists an infinite path with frequency f if and only if there exists a connected set of simple loops having a n.l.c. of ratio f.*

Since the natural linear combination of a set of loops can only have a ratio with rational values, as a corollary we obtain that there does not exist a path with a non-rational asymptotic frequency.

Corollary 3.2.1. *Let G be a k-colored graph, there does not exist an infinite path with frequency f, with a non rational component $f_i \in \mathbb{R} \setminus \mathbb{Q}$, for some $i \in \mathbb{N}$.*

3.2.2 The uniform f-frequency problem

In the previous section we showed that a path with asymptotic frequency f needs to be composed by a periodic repetition of a natural linear composition of simple loops with ratio f and by some other undesired unbalanced contributions that grows sublinearly in the length of the prefix. Since the frequency f is uniform when the expected occurrence of color on a prefix do not differ from the effective ones by a constant, the undesired contributions need to remain bounded and continually balance themselves. Hence, there are no growing unbalanced contributions. Like shown in Lemma 3.2.3, the unbalanced contributions depend on the connection paths between the loops. So, if we want to achieve an uniform frequency, we have to eliminate such connecting paths, by demanding that the loops are not only connected but overlapping as well. Indeed we can prove the following theorem.

Theorem 3.2.2. *In a graph G there exists a path with uniform frequency f if and only if there exists an overlapping set $\mathcal{L}$ of simple loops of G having an n.l.c. of ratio f.*

The proof is a direct consequence of the two following lemmas. The first shows that there exists a path with uniform frequency f if and only if there exists a loop with ratio f, the second lemma shows how this loop can be decomposed in or composed from a set of simple loops with natural linear combination with ratio f. Both lemmas hold for a rational frequency vector $f \in \mathbb{Q}^k$, however, this is not restrictive, since we showed in the previous section no infinite path can have a non-rational frequency component.

Lemma 3.2.4. *Given a graph G and a frequency $f \in \mathbb{Q}^k$, the following statements are equivalent:*

1. *There exists a path with uniform frequency f.*
2. *There exists a periodic path with uniform frequency f.*
3. *There exists a loop with frequency f.*

Proof. 1. $[1 \to 3]$ If $\rho = v_0 a_0 \dots v_n a_n \dots$ is an infinite path with uniform frequency f, then there exists a constant C such that on every prefix the absolute value of the difference between the occurrences vector and the expected occurrences vector is smaller than C. Precisely, for all colors $i \in [k]$ and for all indexes $n \in \mathbb{N}$ we have that $|n \cdot f_i - |\rho^{\leq n}|_i| \leq C$. Let d be the maximum denominator of the rational numbers $f_1, \dots, f_k$, then the vector $n \cdot f - \overline{N_{um}}(\rho^{\leq n})$ belongs to the set of rational vector $\{\frac{x}{y} \in \mathbb{Q} \mid y \leq d, -C \leq \frac{x}{y} \leq C\}^k$. Since both the set of these vectors and the set of nodes in the graph are finite, we can find two indexes $i < j$ such that $v_i = v_j$ and $i \cdot f - \overline{N_{um}}(\rho^{\leq i}) = j \cdot f - \overline{N_{um}}(\rho^{\leq j})$. Consider the loop $\sigma' = v_i a_i v_{i+1} a_{i+1} \dots v_{j-1} a_{j-1} v_j$, its length is $|\sigma'| = j - i$ and its color occurrences vector is $\overline{N_{um}}\sigma' = \overline{N_{um}}(\rho^{\leq j}) - \overline{N_{um}}(\rho^{\leq i}) = f \cdot (j - i)$, hence, the frequency of σ' is exactly f.

2. $[3 \to 2]$ If there exists a loop σ with frequency $\frac{\overline{N_{um}}\sigma}{|\sigma|} = f$, then σ^ω is a periodic path with uniform frequency f. Indeed, every prefix σ^n with $n \in \mathbb{N}$ has color frequency f. So, in the intermediate prefixes, the difference between the color frequency of the prefix and f, never increases beyond the maximum difference in a subpath of σ.

3. $[2 \to 1]$ Trivial

□

Lemma 3.2.5. *Let G be a k-colored graph. There exists a loop with uniform frequency f in G if and only if there exists an overlapping set $\mathcal{L}$ of simple loops of G, with an n.l.c. of ratio f.*

Proof. [only if] If there exists a loop σ with frequency f, by Lemma 1.2.1 the loop is the composition of a tuple $\mathcal{T}$ of simple loops. Let $\mathcal{L}$ be the set of distinct loops occurring in $\mathcal{T}$, and for all $\sigma' \in \mathcal{L}$, let $c_{\sigma'}$ be the number of times σ' occurs in $\mathcal{T}$. Since in the computation of the difference vector of a path it does not matter the order in which the edges are considered, we have $\frac{\sum_{\sigma' \in \mathcal{L}} c_{\sigma'} \cdot \overline{N_{um}}(\sigma')}{\sum_{\sigma' \in \mathcal{L}} c_{\sigma'} \cdot |\sigma'|} = \frac{\overline{N_{um}}(\sigma)}{|\sigma|} = f$. Finally, since the loops in $\mathcal{L}$ come from the decomposition of a single loop σ, we have that $\mathcal{L}$ is overlapping.

[if] Let $\mathcal{L} = \{\sigma_1, \ldots, \sigma_l\}$ be an overlapping set of simple loops such that $\frac{\sum_{i=1}^{l} c_i \cdot \overline{N_{um}}\sigma_i}{\sum_{i=1}^{l} c_i \cdot |\sigma_i|} = f$ By using the algorithm presented in Definition 1.2.9, we construct a loop σ which is a linear composition of $\mathcal{L}$ with coefficients $c_1, \ldots, c_l$. Then, we have that σ has frequency f, since $\overline{N_{um}}(\sigma) = \sum_{i=1}^{l} c_i \cdot \overline{N_{um}}(\sigma_i)$, and $|\sigma| = \sum_{i=1}^{l} c_i \cdot |\sigma_i|$.

□

3.2.3 Relating Priority to Fairness

The graph characterizations for the existence of fair paths are very similar to the characterizations for the existence of f-frequency paths. They both require the existence of a connected or overlapping set of simple loops. The difference is that for the fair path we need a natural linear combination with value zero and for the f-frequency paths we need a natural linear ratio of value f. The following lemma shows that these two properties are related.

Lemma 3.2.6. *Let $\mathcal{L} = \{\sigma_1, \ldots, \sigma_l\}$ be a set of simple loops in a k-colored graph and let $c_1, \ldots, c_l$ be a sequence of coefficients. Then, the natural linear combination of $\mathcal{L}$ with coefficients $c_1, \ldots, c_l$ has value zero if and only if it has ratio $f \in \mathbb{Q}^k$ with $f_i = \frac{1}{k}$ for all $i \in \mathbb{N}$.*

if. If the natural linear combination has ratio with all components equal to $\frac{1}{k}$, then for all colors $j \in [k]$ we have that $\sum_{i=1}^{l} c_i \cdot |\sigma_i|_j = \sum_{i=1}^{l} \frac{c_i}{k} \cdot |\sigma_i| = C$. Hence, for every pair of colors $a, b \in [k]$, we have $\sum_{i=1}^{l} c_i \cdot d_{iff\,a,b}(\sigma_i) = \sum_{i=1}^{l} c_i \cdot (|\sigma_i|_a - |\sigma_i|_b) = 0$.

[only if] If for all pairs of colors $a, b \in [k]$ it holds that $\sum_{i=1}^{l} c_i \cdot d_{iff\,a,b}(\sigma_i) = 0$, then we also have $\sum_{i=1}^{l} c_i \cdot |\sigma_i|_a = \sum_{i=1}^{l} c_i \cdot |\sigma_i|_b$. Hence for all colors $a, b \in [k]$ we have that the ratio $f_a = \frac{\sum_{i=1}^{l} c_i \cdot |\sigma_i|_a}{\sum_{i=1}^{l} c_i \cdot |\sigma_i|} = \frac{\sum_{i=1}^{l} c_i \cdot |\sigma_i|_b}{\sum_{i=1}^{l} c_i \cdot |\sigma_i|} = f_b$. Since $\sum_{j=1}^{k} f_j = 1$ we have $f_j = \frac{1}{k}$ for all $j \in \mathbb{N}$.

□

The above lemma shows that we find a solution to a fairness problem if and only we find a solution to a frequency problem with all equal components. Indeed, the following corollary is an immediate consequence of Lemma 3.2.6, and Theorems 2.2.2, 3.2.1 (resp. Theorems 2.2.1, 3.2.2).

Corollary 3.2.2. *In a k-colored graph there exists a balanced (resp., bounded) path if and only if there exists a path with asymptotic (resp., uniform)frequency with all equal components.*

As one may suspect there is a deeper correlation between fair and priority path. Indeed in the following, we prove that a path is balanced (resp., bounded) if and only if it has asymptotic (resp., uniform) frequency with all equal components.

Theorem 3.2.3. *In a k-colored graph, a path is bounded in difference if and only if it has uniform frequency f with all equal components equal to $\frac{1}{k}$.*

if. If an infinite path ρ has uniform frequency f with all components equal to $\frac{1}{k}$, then there exists a constant C such that for all $n \in \mathbb{N}$ and $i \in [k]$ we have $-C \leq \frac{n}{k} - |\rho^{\leq n}|_i \leq C$. Hence, for all $n \in \mathbb{N}$ and $i, j \in [k]$ we have $d_{\mathit{iff}\,i,j}(\rho^{\leq n}) = (\frac{n}{k} - |\rho^{\leq n}|_j) - (\frac{n}{k} - |\rho^{\leq n}|_i)$. Hence, for all $n \in \mathbb{N}$ and $i, j \in [k]$ we have $-2C \leq d_{\mathit{iff}\,i,j}(\rho^{\leq n}) \leq 2C$, and the path is bounded in difference.

[only if] If an infinite path ρ is bounded in difference , then there exists a constant C such that for all $n \in \mathbb{N}$ and $i, j \in [k]$ we have $-C \leq d_{\mathit{iff}\,i,j}(\rho^{\leq n}) \leq C$. Hence, for all $n \in \mathbb{N}$ and $i, j \in [k]$ we have $d_{\mathit{iff}\,i,j}(\rho^{\leq n}) = (\frac{n}{k} - |\rho^{\leq n}|_j) - (\frac{n}{k} - |\rho^{\leq n}|_i)$. Hence, for all $n \in \mathbb{N}$ and $i \in [k]$ we have $-2C \leq \frac{n}{k} - |\rho^{\leq n}|_i \leq 2C$, and the path is bounded in difference. □

We prove the same result for the balance property and the asymptotic frequency in the next subsection, as a corollary of a more general result.

3.2.4 Limit L problem

Fairness in a balance path in a k-colored graph is achieved by demanding that the differences of colors along the prefixes of the path grow like a function sublinear in the length of the prefix, i.e., for all $i, j \in [k]$ it holds that $\lim_{n\to+\infty} \frac{d_{\mathit{iff}\,i,j}(\rho^{\leq n})}{n} = 0$. Then we can define a priority requirement, by asking that the differences of colors grows like function linear in the length of the prefix, and hence converge to some constants.

Definition 3.2.3. *In a k-colored graph, a path ρ has color limit $L \in \mathbb{R}^{k\times k}$ if for all pairs of colors $i, j \in [k]$ it holds that $\lim_{n\to+\infty} \frac{d_{\mathit{iff}\,i,j}(\rho^{\leq n})}{n} = L_{i,j}$.*

The above problem turns out to be equivalent to the f-frequency problem, as shown by the following lemma.

Lemma 3.2.7. *An infinite path ρ has color limit $L \in \mathbb{R}^{k\times k}$ if and only if it has asymptotic frequency f and the frequency vector f is the unique solution of the following system of $k^2 + 1$ linear equations: for all $i, j \in [k]$, $f_i - f_j = l_{i,j}$; $\sum_{i=1}^{k} f_i = 1$.*

Proof. First, observe that the system of linear equations $f_i - f_j = l_{i,j}$ and $\sum_{i=1}^{k} f_i = 1$ contains k independent rows in the coefficient matrix, i.e., the rows associated with the equations $f_1 - f_k = l_{1,k}, \ldots, f_{k-1} - f_k = l_{k-1,k}$, and $\sum_{i=1}^{k} f_i = 1$. So, the system may have only one solution or no solutions at all.

[only if] If ρ has color frequency vector $f \in \mathbb{R}^k$, then for all $i \in [k]$, it holds that $f_i = \lim_{n\to+\infty} \frac{|\rho^{\leq n}|_i}{n}$. So, for all $i, j \in [k]$, it holds that $l_{i,j} = \lim_{n\to+\infty} \frac{|\rho^{\leq n}|_i - |\rho^{\leq n}|_j}{n} = f_i - f_j$.

[if] If ρ has color limit L then, for all $i, j \in [k]$, it holds that $l_{i,j} = \lim_{n\to+\infty} \frac{d_{iff\,i,j}(\rho^{\leq n})}{n}$. We show that *(i)* $\lim_{j\to+\infty} |\rho^{\leq j}|_k/j = l \triangleq \frac{1-\sum_{a\in[k-1]} l_{a,k}}{k}$; and *(ii)* for all $a \in [k-1]$, the sequence $\{|\rho^{\leq j}|_a/j\}_j$ converges to $l_{a,k} + l$. First we show (i). Assume by contradiction that the sequence is not convergent to l, then we have $\exists \varepsilon > 0\,.\,\forall m \in \mathbb{N}\,.\,\exists n_m \geq m\,.\,\left(\frac{|\rho^{\leq n_m}|_k}{n_m} > l + \varepsilon \text{ or } \frac{|\rho^{\leq n_m}|_k}{n_m} < l - \varepsilon\right)$. The points $\{n_m\}_m$ form a sequence, from which we can extract two subsequences $\{n_{m_i}\}_i$, given by all the points such that $|\rho^{\leq n_{m_i}}|_k/n_{m_i} > l + \varepsilon$, and $\{n_{m'_i}\}_i$, given by all the points such that $|\rho^{\leq n_{m'_i}}|_k/n_{m'_i} < l - \varepsilon$. At least one of the two subsequences is infinite. Assume w.l.o.g. that $\{n_{m_i}\}_i$ is infinite. Then, $\sum_{a=1}^{k-1} l_{a,k} = \sum_{a=1}^{k} \left(\frac{|\rho^{\leq n_{m_i}}|_a}{n_{m_i}} - l\right) > \sum_{a=1}^{k-1} \left(\frac{|\rho^{\leq n_{m_i}}|_a}{n_{m_i}} - l\right) + \varepsilon$. In other words, $\sum_{a=1}^{k-1} \left(\frac{|\rho^{\leq n_{m_i}}|_a}{n_{m_i}} - l\right) < \sum_{a=1}^{k-1} l_{a,k} - \varepsilon$. So, for all $i \in \mathbb{N}$ there is a color $a \in [k-1]$ such that $\frac{|\rho^{\leq n_{m_i}}|_a}{n_{m_i}} - l \leq l_{a,k} - \frac{\varepsilon}{k-1}$. Then, there is a color $a \in [k-1]$ and a subsequence $\{n_{m_i^a}\}_i$ of $\{n_{m_i}\}_i$ such that for all $i \in \mathbb{N}$ we have that $\frac{|\rho^{\leq n_{m_i^a}}|_a}{n_{m_i^a}} < l_{a,k} + l - \frac{\varepsilon}{k-1}$. Moreover, for all $i \in \mathbb{N}$ we have $\frac{d_{iff\,a,k}\left(\rho^{\leq n_{m_i^a}}\right)}{n_{m_i^a}} = \frac{\left|\rho^{\leq n_{m_i^a}}\right|_a - \left|\rho^{\leq n_{m_i^a}}\right|_k}{n_{m_i^a}} \leq \left(l_{a,k} + l - \frac{1}{k-1}\varepsilon\right) - (l + \varepsilon) = l_{a,k} - \frac{k}{k-1}\varepsilon$. Therefore, the sequence $\{d_{iff\,a,k}(\rho^{\leq n_{m_i^a}})/n_{m_i^a}\}_i$ does not converge to $l_{a,k}$, so does not the sequence $\{d_{iff\,a,k}(\rho^{\leq j})/j\}_j$, since the first is a subsequence of the latter. So, by contradiction, we have proved (i).

Now, we show (ii). Assume by contradiction that $\{|\rho^{\leq j}|_a/j\}_j$ does not converge to $l + l_{a,k}$, for a certain $a \in [k-1]$. Then, we have

$$\exists \varepsilon > 0\,.\,\forall m \in \mathbb{N}\,.\,\exists n_m \geq m\,.\,\left(\frac{|\rho^{\leq n_m}|_a}{n_m} > l + l_{a,k} + \varepsilon \text{ or } \frac{|\rho^{\leq n_m}|_a}{n_m} < l + l_{a,k} - \varepsilon\right). \quad (3.2)$$

Let ε be a witness for (3.2). By (i), there is $\bar{n} \in \mathbb{N}$ such that for all $n \geq \bar{n}$, we have $l - \varepsilon/2 < |\rho^{\leq n}|_k/n < l + \varepsilon/2$. So, for all $m \geq \bar{n}$, there is $n_m \geq m$ such that either *(a)* $\frac{|\rho^{\leq n_m}|_a}{n_m} > l + l_{a,k} + \varepsilon$ or *(b)* $\frac{|\rho^{\leq n_m}|_a}{n_m} < l + l_{a,k} - \varepsilon$, depending on which disjunction in (3.2) holds. Assuming that *(a)* occurs for infinitely many n_m, for all $m \geq \bar{n}$ there is $n_m \geq m$ such that

$$\frac{d_{iff\,a,k}(\rho^{\leq n_m})}{n_m} = \frac{|\rho^{\leq n_m}|_a - |\rho^{\leq n}|_k}{n_m} > l + l_{a,k} + \varepsilon - \left(l + \frac{\varepsilon}{2}\right) = l_{a,k} + \frac{\varepsilon}{2}.$$

Thus, we have that $\{d_{iff\,a,k}(\rho^{\leq n})/n\}_n$ does not converge to $l_{a,k}$, which is a contradiction. □

It i s easy to observe that a path is balanced if and only if it has color limit $L = \mathbf{0}$, since the two definition coincides for $L = \mathbf{0}$. Hence, the following corollary follows.

Theorem 3.2.4. *In a k-colored graph, a path is balanced if and only if it has asymptotic frequency f with all equal components equal to $\frac{1}{k}$.*

Proof. A path is balanced if and only if it has color Limit $\mathbf{0}$, by Lemma 3.2.7 this happens if and only if it has asymptotic frequency f, whose components are the only solution of the system of k^2+1 linear equations: for all $i,j \in [k]$, $f_i - f_j = 0$; $\sum_{i=1}^{k} f_i = 1$. The only solution f has all equal components equal to $\frac{1}{k}$. □

3.3 Solving the frequency-f problem

Since the graph characterization for the existence of a path with asymptotic frequency f is very similar to the graph characterization for the existence of a balanced path, the algorithms for the solution of the two problems are similar as well.

In Section 2.3, we determine a connected set of simple loops with zero as a value of a natural linear combination by means of a system of linear inequalities. There is a variable for each edge, representing how many times that edge is used in the natural linear combination of those loops. Moreover, a part of the constraints ensure that the edges actually form loops and the other part ensure that the natural linear combination has value zero. The connection between the loops is ensured by making the algorithm run only on connected components.

In order to determine a path with asymptotic frequency $f \in \mathbb{Q}^k$, we need to determine a connected set of simple loops with a natural linear ratio of value f. Hence, we can use the same system for the balance problem, we just need to substitute the constraints about the natural linear combination of value zero with a new set of constraints ensuring the existence of a natural linear combination of ratio f. Since for each $e \in E$, the variable x_e represent how many times the edge e is used in the linear combination, we have that $\sum_{e \in E(a)} x_e$ is the number of edges with color a in the linear combination and $\sum_{e \in E} X_e$ is the total number of edge. Hence, the natural linear ratio is f, if and only if for all colors $a \in [k]$ it holds that $\frac{\sum_{e \in E(a)} x_e}{\sum_{e \in E} x_e} = f_a$

Definition 3.3.1. *Let $G = (V, E)$ be a k-colored graph, and $f = (\frac{x_1}{y_1}, \ldots, \frac{x_k}{y_k}) \in \mathbb{Q}^k$ a rational vector. We call* real asymptotic frequency-f system *for G the following system of equations on the set of variables $\{x_e \mid e \in E\}$.*

1. for all $v \in V$	$\sum_{e \in v^{\leftarrow}} x_e$	$= \sum_{e \in v^{\rightarrow}} x_e$
2. for all $a \in [k]$	$x_a \sum_{e \in E(a)} x_e$	$= y_a \sum_{e \in E} x_e$
3. for all $e \in E$	x_e	≥ 0
4.	$\sum_{e \in E} x_e$	$= 1$
5.	x_e	$\in \mathbb{R}$.

Let $m = |E|$ and $n = |V|$, the frequency-f system has m variables and $m+n+k^2+1$ constraints.

Like we did in Section 2.3, we can define a natural frequency-f system which differs from the above in two points: *(i)* The fourth constraint is $\sum_{e \in E} x_e > 0$ and *(ii)* the fifth constraint asks that the variable x_e are natural numbers. Using a very similar proofs to those developed for Lemmas 2.3.2 and 2.3.3 we prove respectively the following two lemmas.

Lemma 3.3.1. *There exists a set $\mathcal{L}$ of simple loops in G with an n.l.c. of ratio $f \in \mathbb{Q}^k$ if and only if the natural asymptotic frequency-f system for G is feasible.*

Lemma 3.3.2. *There exists a solution to the real asymptotic frequency-f system if and only if there exists a solution to the natural asymptotic frequency-f system.*

Observe that the proofs just change only in the part dealing with the second set of constraints: instead of proving that these constraints are equivalent to a value zero for the natural linear combination, we prove instead that they are equivalent to a ratio f for the n.l.c. Since this part is just a matter of algebra, we leave the modification of the proof to the reader. From these two lemmas we obtain the following corollary.

Corollary 3.3.1. *If G is strongly connected, there exists an asymptotic frequency-f path in G if and only if the real asymptotic frequency-f system for G is feasible.*

At this point we can develop an algorithm for determining a representation of a frequency-f path. This algorithm is similar to the one for the determination of a balanced path.

Theorem 3.3.1. *The frequency-f problem, i.e. determining whether on a graph G there exists a path with asymptotic frequency $f \in \mathbb{Q}^k$ and computing a representation of such path if it exists, is polynomial in the size of G and the logarithm of the maximum number $d \in \mathbb{N}$ appearing as nominator or denominator of a rational component of f.*

Proof. In order to solve the frequency-f problem in G, first we compute the maximal connected components of G using the classical Tarjan's algorithm [CLRS01]. This algorithm is polynomial in n and m. Then, in each component we compute whether the real balance system is feasible, by using the polynomial algorithm for feasibility of sets defined by linear constraints [NW88]. This second algorithm is used at most n times and it is polynomial in the number of constraints $(n + m + k)$ and in the logarithm of the maximum modulus of a coefficient in a constraint (in our case, the maximum modulus is d). If the feasibility algorithm answer positively to the existence of a solution, it also provides a rational one [NW88]. By the proof of Lemma 3.3.1, such a solution allows us to compute in polynomial time a set of connected simple loops and the coefficients of an n.l.c. of ratio f. As shown in the *if* part of the proof of Lemma 3.2.3, this in turn allows us to constructively characterize a balanced path in the graph.

□

3.4 Solving the uniform frequency-f problem

Since the graph characterization for the existence of a path with uniform frequency f is very similar to the graph characterization for the existence of a bounded path, the algorithms for the solution of the two problems are similar as well.

Both problems are equivalent to the existence of a certain overlapping set of simple loops: the bounded problem requires that these loops have a natural linear combination with value zero, and the uniform frequency-f problem requires that these loops have a natural linear combination of ratio f. Hence, one may imagine that, like it was done for the asymptotic frequency f problem, it

is sufficient to change the second set of constraints of the bounded linear system in such a way to require that variables represent loops with natural linear combination of ratio f, instead of a natural linear combination of value zero. However, the change in the constraints about the variables x_e, may not ensure the validity of a lemma like Lemma 2.4.1 for the real frequency-f system. Hence, the ninth constraint of the bounded path problem, i.e. $y_e \leq (m \cdot s_G!)x_e$, may avoid the possibility to find some solutions. Indeed, for a set of overlapping simple loops satisfying the constraints of the real f-frequency problem, the connection flow (equal to $\sum_{e \in E} x_e$) may be larger than $(m \cdot s_G!)x_e$ In such a case with the ninth constraint, we do not find a connection flow and the solution does not belong to those found by the linear system. Hence, we need a new upper bound on the maximum value of $\sum_{e \in E} x_e$ and on the minimum value of a variable x_e. The following lemma answer to the question.

Lemma 3.4.1. *Let $G = (V, E)$ be a k-colored graph, with $|V| = n$, $|E| = m$, and $s_G = min\{n + k - 1, m\}$. Let $f \in \mathbb{Q}^k$ be a frequency vector and $d \in \mathbb{N}$ be the maximum number appearing as a denominator or nominator among the components of f. For all solutions x to the real balance system for G there exists a solution x' such that, for all $e \in E$, it holds $(x_e = 0 \Rightarrow x'_e = 0)$ and $(x_e > 0 \Rightarrow 1 \leq x'_e \leq s_G!)$. As a consequence, $1 \leq \sum_{e \in E} x'_e \leq m \cdot s_G! \cdot d^{s_G}$.*

Proof. Let x be a solution to the frequency-f system for G, and let J be the set of all edges e such that $x_e > 0$. By construction, $|J| > 0$. We represent the first two sets of equalities of the frequency-f system in matrix form as $Dx = \mathbf{0}$. Then, the set of points satisfying the frequency-f system is $P = \{y \in \mathbb{R}^m \mid Dy = \mathbf{0}, y \geq \mathbf{0}, \sum_{e \in E} y_e > 0\}$. Now the subset of P, $P' = \{y \in P \mid \forall e \in J \,.\, y_e \geq 1 \text{ and } \forall e \notin J \,.\, y_e = 0\} = \{y \in P \mid \forall e \in E \,.\, (x_e > 0 \Rightarrow y_e > 1) \text{ and } (x_e = 0 \Rightarrow y_e = 0)\}$ is not empty. Indeed, the vector $z = x(\min_{e \in J} x_e)^{-1}$ is in P', since (i) $Dz = (\min_{e \in J} x_e)^{-1} Dx = \mathbf{0}$, (ii) for all $e \in J$, we have $z_e = x_e(\min_{e \in J} x_e)^{-1} \geq 1$, and (iii) for all $e \notin J$, we have $z_e = 0$.

The set of inequalities "$\forall e \in J \,.\, y_e \geq 1$" can be represented as the system $Fy \geq \mathbf{1}$, with $\mathbf{1} \in \{1\}^{l \times 1}$. Similarly, the set of equalities "$\forall e \notin J \,.\, y_e = 0$" can be represented as $F'y = \mathbf{0}$. If we define $D' = \binom{D}{F'} \in \{-1, 0, 1\}^{(2n+k-l-1) \times m}$, we have $P' = \{y \in \mathbb{R}^m \mid D'y = \mathbf{0}, Fy \geq \mathbf{1}\}$. Since $D', F, \mathbf{1}, \mathbf{0}$ all have elements in $\{-d, -(d-1) \ldots -1, 0, 1 \ldots d-1, d\}$, by Lemma 2.4.4 the set P contains an element $x' \in \mathbb{Q}^m$ such that for all $i \in [m]$, $x'_i \leq (\min\{2n + k - 1, l + m\})! \cdot d^{\min\{2n+k-1, l+m\}} \leq (\min\{2n + k - 1, n + m\})! \cdot d^{\min\{2n+k-1, l+m\}} = s_G! \cdot d^{s_G}$, which concludes the proof. □

The lemma allows us to state that whenever there exists a solution to the asymptotic frequency-f problem, then there exists also another solution x_e such that $\sum_{e \in E} x_e \leq s_G! \cdot d^{s_G}$ and $x_e \geq 1$. Hence, a connection flow y_e does not need to be greater than $m \cdot s_G! \cdot d^{s_G} \cdot x_e$.

By using the above constraint on the connection flow in place of the ninth constraint of the bounded system, and by using the asymptotic frequency-f constraints in place of the balanced constraints, from the bounded system we obtain the uniform frequency-f system.

Definition 3.4.1. *Let $G = (V, E)$ be a k-colored graph with $m = |E|$, $n = |V|$, and $s_G = \min\{n + k - 1, m\}$, let $u \in V$ be a node, and $f \in \mathbb{Q}^k$ be a frequency vector with $d \in \mathbb{N}$ equal to the maximum number appearing as a denominator or nominator among the components of*

f. We call uniform frequency-f system for (G, u) the following system of equations on the set of variables $\{x_e, y_e \mid e \in E\}$.

1-4. The same constraints as in the balance system for G		
5. for all $v \in V \setminus \{u\}$	$\sum_{e \in E_v} y_e - \sum_{e \in_v E} y_e$	$= \sum_{e \in_v E} x_e$
6.	$\sum_{e \in_u E} y_e - \sum_{e \in E_u} y_e$	$= \sum_{v \in V \setminus \{u\}} \sum_{e \in_v E} x_e$
7. for all $e \in E$	y_e	≥ 0
8. for all $e \in E$	y_e	$\leq (m \cdot s_G! \cdot d^{s_G}) x_e$.

The bounded difference system has $2m$ variables and $3m + 2n + k$ constraints.

The following lemma states that the bounded difference system can be used to solve the bounded difference problem.

Lemma 3.4.2. *There exists an overlapping set of simple loops in G, passing through a node u and having an n.l.c. of ratio f if and only if the uniform frequency-f system for (G, u) is feasible.*

The proof of the lemma is the equal to the proof of Lemma 2.4.2. We just need to use Lemmas 3.4.1 and 3.3.1 in place of Lemmas 2.4.1 and 2.3.2 respectively.

At this point, computing a rational solution x to the real frequency-f system provides us with a set of overlapping loops representing a bounded path in a k-colored graph. On the other hand, if we do not find such a solution we conclude that there are no loops with f as ratio of an n.l.c. and, hence, no path with uniform frequency f.

Theorem 3.4.1. *Them uniform frequency-f problem, i.e. determining whether on a graph G there exists a path with uniform frequency f and computing a representation of such path if it exists, is polynomial in the size of G and f*

Proof. In order to solve the bounded difference problem in G, for all $u \in V$ we check whether the bounded difference system for (G, u) is feasible, by using a polynomial time algorithm for feasibility of linear systems [NW88]. This algorithm is used at most n times and it is polynomial in the number of constraints ($2n + 3m + k$) and in the logarithm of the maximum modulus M of a coefficient in a constraint. In our case, $M = m \cdot s_G! \cdot d^{s_G}$. Using Stirling's approximation, we have $\log(m \cdot s_G!) = \log(m) + \Theta(s_G \log(s_G))$. Therefore, we obtain the following.

If the feasibility algorithm answer positively to the existence of a solution, it also provide a rational one [NW88]. By the proof of Lemma 3.3.1, the component x of the solution allows us to compute in polynomial time a set of simple loops and the coefficients of an n.l.c. of ratio f. By the properties ensured by the component y of the solution we know that the loops are overlapping. As shown in the *if* part of the proof of Lemma 3.2.5, this in turn allows us to constructively characterize a uniform frequency-f path in the graph. □

3.5 Problems on Initialized Graphs

In general, a system may have a starting state, in which case it makes sense to determine a fair or priority path reachable from the graph's starting node. Since a fair or priority path is constructed from a set of simple loops, in order to ensure that path is reachable from the starting node, we

have to ensure that the loops are reachable from this node. As long as balanced and asymptotic frequency-f paths are considered, we solve the relative problems by looking for loops in some strongly connected components of the graph. Hence, to make sure the loops are reachable from the node, we just consider those connected components which are reachable from the starting node. On the other hand for bounded in difference and uniform frequency-f path we use just one linear system for the whole graph. At this point the problem may be solved in two ways. *(i)* we decompose the graph in its connected components and then we look for a set of overlapping loops only on those component that are reachable from the starting node. *(ii)* we add to the bounded or uniform frequency-f linear system a set of constraints ensuring that the loops are reachable from the starting node. In the same way we ensured the reachability of all the nodes of the loops from a given node u, we can define a second connection load z_e for each edge e. Such connection load should be generated by the starting node and all the nodes of the loops should absorb some positive amount of it. To this aim, we ask that each node v should absorb a load equal to $\sum_{e \in v^{\rightarrow}} x_e$, and that the starting node should produce a load equal to $\sum_{v \in V} \sum_{e \in v^{\rightarrow}} x_e = \sum_{e \in E} x_e$. Since such connection load can use arbitrary any edge of the graph, we have no special requirement for the load z_e. Hence, the set of connection constraint to the starting node v_{ini} are given by the following system.

Definition 3.5.1. *Let $G = (V, E)$ be a k-colored graph with $m = |E|$, $n = |V|$. The following is the set of constraints to add to the bounded system or to the uniform frequency-f system in order to ensure the reachability of the loops from a starting node v_{ini}.*

$$
\begin{array}{lrcl}
\textit{1. for all } v \in V \setminus \{u\} & \sum_{e \in v^{\leftarrow}} z_e - \sum_{e \in v^{\rightarrow}} z_e & = & \sum_{e \in v^{\rightarrow}} x_e \\
\textit{2.} & \sum_{e \in v_{ini}^{\rightarrow}} z_e - \sum_{e \in v_{ini}^{\leftarrow}} z_e & = & \sum_{e \in E} x_e \\
\textit{3. for all } e \in E & z_e & \geq & 0 \\
\textit{4. for all } e \in E & z_e & \in & \mathbb{R}.
\end{array}
$$

Checking that a connected component is reachable from a starting node, running the bounded or uniform f-frequency algorithm as many times as the number of connected components, and adding a polynomial number of constraints to the feasibility systems, are the only operation needed to solve the initialized version of the problems. Hence, all these problems are still solvable in polynomial time in the size of the graph and of the frequency components.

In the next section we prove that the discussed fairness and priority properties are prefix-independent, i.e., given a a fair of priority path ρ, a new path obtained from ρ by removing or adding a finite prefix still satisfies the same fairness or priority property or ρ. Hence, once we find a fair or priority path ρ reachable from a starting node v_{ini} for any finite path π connecting v_{ini} to a node $\rho(n)$, we have that $\pi \cdot \rho^{>n}$ is a path starting from ρ and satisfying the same fairness or priority property of ρ.

3.6 Discussion

In a colored graph a path has asymptotic frequency f if for all colors i the difference between f_i and the frequency of a prefix of the path grows sublinearly in the length of the prefix. When the difference does not grow and it is bounded by a constant the frequency is called uniform. Since, the

uniform frequency-f property is derived by the more general asymptotic frequency-f property by choosing a particular sublinear function that bounds the difference between the frequency of colors on the prefixes and f, we may think to define for each sublinear function $g(n) \in o(n)$ bounding the difference a $g(n)$-asymptotic frequency-f property. Depending on how slowly $g(n)$ grows we have a different degree of uniformity of the prefix along the path.

However, we characterized the existence of an asymptotic frequency-f path by means of a connected set of simple loops with an n.l.c. of ratio f, and the existence of an uniform frequency-f path by means of an overlapping set of simple loops with an n.l.c of ratio f. What would be a middle point between connection and overlapping of loops, that allows us to achieve a $g(n)$-asymptotic frequency but not yet an uniform frequency?

Observe that when we determine an n.l.c $\{(\sigma_1, c_1), \ldots, (\sigma_l c_l)\}$ of ratio f and paths π_j connecting σ_j to $\sigma_{(j+1 \mod l)}$, we can construct the $\rho = \prod_{i=1}^{+\infty} \rho_i$ where $\rho_i = (\prod_{j=1}^{l} (\sigma_j)^{i^d} \cdot \pi_j)$. Following a proof similar to the one of Lemma 3.2.3, one can prove that ρ has asymptotic frequency f. Also it is possible to prove that the difference between f_i and $|\rho^{\leq n}|_i$ grows as $C \cdot n^{-\frac{1}{d+1}}$. Indeed, every time a path ρ_i ends, the length is $C_1 \cdot i^{d+1}$, and the difference of occurrences of colors from the expected $f_i \cdot n$ is $|i \cdot \sum_{j=1}^{l} |\sigma_j|_i| \leq C_2 \cdot i$. In the intermediate points from ρ_i to ρ_{i+1} the difference of occurrences of colors from the expected $n \cdot f_i$ is bounded by $C_2 \cdot (i+1) + |\sum_{j=1}^{l} |\sigma_j|_i^{i^d}| \leq C_3 \cdot i^d$. Hence, the difference of frequency grows like $\frac{1}{i} = n^{-\frac{1}{d+1}}$

Hence, whenever we determine a path with asymptotic frequency-f we also determine a path with $n^{-\varepsilon}$-asymptotic frequency f for every $\varepsilon > 0$. We hardly believe we can find a intermediate level of connection of loops between path-connection and overlapping, that can allow a non-bounded path with better uniformity than any $n^{-\varepsilon}$. Our conjecture is that one cannot perform better without obtaining a bounded path, i.e. for all $g(n) \in \cap_{\varepsilon > 0} o(n^{-\varepsilon})$ there exists a $g(n)$-asymptotic frequency-f path if and only if there exists a bounded difference path.

Part II

Problems on Arenas

...

...

4

Games for scheduling

Contents

4.1 Introduction

In this part of the thesis we extend the model of colored graph to colored games in order to encapsulate also sources of non-determinism in a scheduling plan. Hence, the activity graph is endowed with a partition, distinguishing between the system's states controlled by the scheduler and the system's states controlled by the environment. This part is divided in two chapters. In the first we show that determining whether the system is able to follow a path satisfying a fairness or priority condition defined in the Part 1 is a $Co - NP$-hard problem. Fundamental in the proof of this result is the property of half-positionality for our fairness and priority goals. Since, this property may allow to prove a similar result for other game goals, maybe other fairness and priority goals, we analyze it in the second chapter. There we define a novel sufficient condition for half-positionality, broader than the one already known in literature.

Preliminaries on Games

Like we defined in the preliminaries of this thesis, a colored arena is an initialized colored graph endowed with a partition V_0, V_1 of the set of nodes. We imagine that on an arena two players, called 0 and 1, alternate moves and construct a path. Every time a partial path is on a node of the set V_i, player i decides what edge is added next. In our framework, player 0 represents the system or scheduler, meanwhile player 1 represents the environment.

Definition 4.1.1. k-colored arena

1. *A k-colored arena is a structure (V_0, V_1, E) where $(V_0 \cup V_1, E)$ is a k-colored graph, and the two sets V_0, V_1 are disjoint.*

2. *An initialized k-colored arena is a structure (V_0, V_1, E, v_{ini}) where $(V_0 \cup V_1, E, v_{ini})$ is an initialized k-colored graph, and the two sets V_0, V_1 are disjoint.*

3. *We call the nodes of a set V_i, nodes belonging to player i.*

4. *The set of edges starting from nodes of player i is denoted as $E_i^{\leftarrow}$, the set of edges ending in nodes of player i is denoted as $E_i^{\rightarrow}$.*

5. *A finite path starting from a node v in an arena, is called finite or partial v-play, because it is an intermediate result of the two players alternating moves.*

6. *An infinite path starting from a node v in an arena, is called v-play (or infinite play), because is the definitive result of the two players alternating moves*

A rule, describing what edge a player chooses at every partial play, is called a strategy for that player. Since a play is determined, once the two player have established their strategies, the analysis of these rules provides insights about the ability of the players to pursue their goals.

Definition 4.1.2. Strategy

1. *A strategy for player $i \in \{0,1\}$ is a function $\tau_i : (V_0 \cup V_1) \cup E^+ \to E$ that associates an edge-choice to each partial play and to each starting node. Observe, that player i just needs to decide an edge at every partial play ending with a node of player i and at each starting node belonging to player i. However, this more general form of strategy makes our formal reasoning easier and does not make a strategy's construction more complex because the unused values can be set arbitrary.*

2. *A partial play or play ρ starting at a node v is said consistent with a strategy τ_i for player i, if* (i) *the starting edge of ρ is $\tau_i(v)$ and* (ii) *for every non-empty prefix ρ' of ρ ending with a node in V_i, the edge of the play following ρ' in ρ is the one computed by the strategy, i.e., $\tau_i(\rho')$. The empty path is considered consistent with any strategy.*

3. *Given two strategies τ_0 and τ_1 for the two players, for every node $v \in V_0 \cup V_1$, there exists only one play starting at v and consistent with τ_0 and τ_1. Indeed the first edge is chosen by the strategy of the player i to which v belongs and is $\tau_i(v)$. Inductively at every other partial play the next edge is chosen by the strategy of the player the ending node belongs to.*

4. *A strategy τ_i of player i is said memoryless if the choice of player i at every partial play depends only on the last node of the play. Formally τ_i is memoryless if there exists a function $\tau_i' : V_i \to E$ such that for all partial paths ρ ending with $v \in V_i$ we have $\tau_i(\rho) = \tau_i'(v)$. In such a case, τ_i' is a synthetic way to represent τ_i and is considered as a strategy too.*

The two player alternate moves and construct an infinite path, when one player seeks the satisfaction of a give path property and the other player pursue the complementary goal, then we have a game between them. In this thesis the goal of player 0 is the system specification the scheduler aims to satisfy, meanwhile player 1, depicted as a malicious entity, tries to do the opposite.

Definition 4.1.3. Games

1. *A goal or winning condition W on a k-colored arena, is a set of infinite sequences of colors in $[k]$, i.e., $W \subseteq [k]^\omega$.*

2. *A k-colored game is a structure $G = (A, W)$ composed by an initialized k-colored arena $A = (V_0, V_1, E, v_{ini})$ and a k-colored winning condition W.*

3. *A path on A with color sequence in W is said winning for player 0, a path on A with color sequence out of W is said winning for player 1.*

4. *A strategy τ_i for player i is said winning for player i at a node $v \in V_0 \cup V_1$, if every play starting at v and consistent with τ_i is winning for player i.*

5. *The winning set of a strategy τ_i for player i is a set V' of all and only the nodes $v \in V_0 \cup V_1$, such that τ_i is winning at the node v.*

6. *A game is said winning for player i if there exists a winning strategy for player i at the starting node v_{ini}.*

A winning strategy for player 0 represents a rule the scheduler can follow in order to ensure that the sequence of planned activity satisfies the given system specification, no matter what the environment does. On the other hand, a winning strategy for player 1 represents a way for the environment to force a run that fails to satisfy the given property, hence, the non-existence of a valid plan for the scheduler. Also when a winning strategy for player 0 is memoryless, the scheduling plan is easy, since the activity the scheduler has to chose at every state depends only on the state and not on some history of the run. Hence, it is interesting to relate a game to the existence of winning strategy and memoryless strategy.

Definition 4.1.4. Positionality and Determinacy

1. *A game is said determined if one of the two players has a winning strategy.*
2. *A determined game is said positional if the winning player has a memoryless winning strategy.*
3. *A goal W is said determined if all games with goal W are determined.*
4. *A goal W is said full-positional if all determined games with goal W are positional*
5. *A goal W is said half-positional for player i, if all games having goal W and winning for player i, are positional.*

4.1.1 Strategy and Memory

In general, a strategy allows a player to compute the edge that needs to follow a partial play. It is possible that such a computation does not need to evaluate the whole path, but maybe a more succinct information extracted from the path itself. This information represents what a player needs to remember in order to make a choice and it is called memory.

Definition 4.1.5. Memory

Consider an arena $A = (V_0, V_1, E)$ and a strategy $\tau : (V_0 \cup V_1) \cup E^+ \to E$ for player i. A memory of τ is a function $\mu : (V_0 \cup V_1) \cup E^+ \to M$ such that for all paths $\rho, \rho' \in (V_0 \cup V_1) \cup E^+$ with $\mu(\rho) = \mu(\rho')$ we have $\tau(\rho) = \tau(\rho')$. The set M is called memory space.

Observe that every strategy has a memory, indeed the identity function $\mu(\rho) = \rho$ for all paths and starting node is a valid memory for any strategy. If $\mu : (V_0 \cup V_1) \cup E^+ \to M$ is memory function for τ, then we can define a function $\tau_\mu : M \to E$ such that for all $m \in M$ we have $\tau_\mu(m) = \tau(\rho)$ for every path or starting node ρ with memory m. Since τ is the composition of τ_μ with μ, we have that the triple (M, μ, τ_μ) is an alternative way to represent the strategy τ, and it is called memory form of τ.

The use of memory becomes particularly interesting, when there is an updating function $\psi : M \times E \to M$ that allows to determine what is the memory $\psi(m, e)$ of a path obtained by adding an edge e to a prefix with known memory m.

Definition 4.1.6. Update function

Consider an arena $A = (V_0, V_1, E)$, and a strategy (M, μ, τ_μ) for player i in memory form with $\mu : (V_0 \cup V_1) \cup E^+ \to M$ and $\tau : M \to E$. Then, $\psi : M \times E \to M$ is an updating function, if for all paths ρ and edges e we have $\mu(\rho \cdot e) = \psi(\mu(\rho), e)$.

If an update function exists, then the strategy (M, μ, τ_μ) for player i can be represented as a quadruple $(M, \psi, \tau_\mu, \mu_0)$ where τ_μ is the choice function, ψ is the update function and $\mu_0 : (V_0 \cup V_1) \to M$ is the initial memory function such that $\mu_0(v) = \mu(v)$ for all $v \in V_0 \cup V_1$. Indeed, player i needs just to update its memory every time and edge is added to the play, and when the play is on a node belonging to player i, and the memory is m, then player i chooses the next edge $\tau_\mu(m)$. A strategy in memory-update form can easily be implemented, since at every step the player just needs to compute the value of two functions. However, for implementation purposes we need an algorithm for the computation of ψ and τ. Observe that, whenever M is finite, the two function ψ and τ can always be given as tables (an $|M||E| \times |M|$ table for ψ and an $|M| \times |E|$ table for τ). In such case $|M|^2 \cdot |E|$ is the size of the strategy.

Observe that given a strategy $(M, \psi, \tau_\mu, \mu_0)$ in memory update form, we can compute its memory form (M, μ, τ_μ), indeed $\mu(v) = m_0(v)$ and for all paths $\pi = e_1 \dots, e_n$ we have $\mu(\pi) = \psi(\psi(\dots \psi(\psi(m_0, e_1), e_2) \dots), e_n)$.

Definition 4.1.7. *Unfolding Function*

For every update function $\psi : M \times E \to M$, we call unfolding of ψ, the function $\psi^* : M \times E^* \to M$ such that

1. *$\psi^*(m, \varepsilon) = m$,*
2. *for all $\pi = e_1 \dots e_n$ of length $n \geq 1$ we have that the unfolding function is equal to $\psi^*(m, \pi) = \psi(\psi(\dots \psi(\psi(m, e_1), e_2) \dots), e_n)$.*

For every memory function $\mu : 0 : (V_0 \cup V_1) \to M$, we call fixed unfolding of ψ, the function $\psi^*[m] : E^* \to M$ such that for all $\pi \in E^*$ with starting node v, we have $\psi^*[m](\pi) = u^*(\mu_0(v), \pi)$

It is easy to see that the memory function for a strategy $(M, \psi, \tau_\mu, \mu_0)$ is $\mu := \psi^*[m]$. It is immediate to prove from the definition of unfolding of ψ the following concatenation property.

Lemma 4.1.1. *For every path $\pi = \pi_1 \cdot \pi_2 \in E^*$, and for every memory m we have $\psi^*[m](\pi) = \psi^*(m, \pi) = \psi^*(\psi^*(m, \pi_1), \pi_2) = \psi^*[\psi^*[m](\pi_1)](\pi_2)$.*

4.1.2 Fairness and Priority Goals

In this chapter we focus our attention on games with the Fairness and Priority goals introduced respectively in Chapter 2 and Chapter 3.

1. The *balance* goal W_{bl} is the set containing all and only the balanced color sequences.
2. The *bounded* goal W_{bn} is the set containing all and only the bounded color sequences.
3. Let $f \in \mathbb{Q}^k$ be such that $\sum_{i=1}^k f_i = 1$. The *asymptotic frequency-f* goal W_{af} is the set containing all and only the color sequences with color asymptotic frequency vector f.

4. Let $f \in \mathbb{Q}^k$ be such that $\sum_{i=1}^{k} f_i = 1$. The *uniform frequency-f* goal W_{uf} is the set containing all and only the color sequences with color uniform frequency vector f.

Observe that, due to Corollary 3.2.1, we investigate only priority goals with a rational frequency. Given a frequency vector $f \in \mathbb{Q}^k$ with all components equal to $\frac{1}{k}$, by Theorems 3.2.4 and 3.2.3 we obtain that $W_{bl} \subseteq W_{af}$ and $W_{bn} \subseteq W_{uf}$.

4.2 Preliminaries on Half-positionality

Half-positionality states that the winning capability of a player is not weakened if we make it only use memoryless strategies, since, when that player wins, it wins with a memoryless strategy. Half-positionality is used in the course of the first chapter to prove results on games. Hence, as a preliminary we now discuss, a sufficient condition due to Erik Kopczyński [Kop06]. In the following, unless otherwise stated, we just use the term half-positionality, to mean half-positionality for player 0.

4.2.1 Kopczyński's theorem

In [Kop06], Kopczyński defines two properties of goal, sufficient for half-positionality. A desirable property is that every time the play passes in a node controlled by player 0, this player always prefer progressing through the same edge rather than alternating between different ones. Kopczyński captures this property through the concept of concavity, which states that no matter how two losing words are interleaved the results is a losing word.

Definition 4.2.1. Shuffle

The shuffle of two infinite words $x, y \in [k]^\omega$ is the set of infinite words obtained by switching infinitely often between x and y, i.e., $x, y\otimes = \{z_1 \dots z_n \dots \in [k]^\omega \mid z_1 z_3 \dots z_{2n+1} \dots = x, z_2 z_4 \dots z_{2n} \dots = y\}$.

Definition 4.2.2.

A goal $W \subseteq [k]^\omega$ is concave, if for each pair of losing infinite words $x, y \notin W$, all words in the shuffle are losing, i.e., $x, y \otimes \cap W = \emptyset$.

A goal $W \subseteq [k]^\omega$ is convex, if its complement $[k]^\omega \setminus W$ is concave

Lemma 4.2.1. *A goal $W \subseteq [k]^\omega$ is convex if and only if for each pair of winning infinite words $x, y \notin W$, all words in the shuffle are winning, i.e., $x, y\otimes \subseteq W$.*

Every time player 0 reaches a node with multiple exiting edges, he has a choice between many positional behaviors. If one of them is winning, player 0 uses that one. If they are all losing, concavity ensures that they cannot be alternated in order and give a non-memoryless winning behavior. However, it is still possible that the optimal positional choice of player 0 depends on some finite prefix up to that node. Kopczyńsky solves the problem through the property of prefix-independence, which states that the winning value of a word does not change whether we modify some finite prefix.

Definition 4.2.3. *A goal W is prefix-independent if, for all infinite words $x \in [k]^\omega$, and for all finite words $z \in [k]^*$, the word x is winning if and only if the word $z \cdot x$ is winning as well.*

Lemma 4.2.2. *A goal $W \subseteq [k]^\omega$ is prefix-independent if and only if its complement $[k]^\omega \setminus W$ is prefix independent.*

Together, the two properties constitute a sufficient condition not only to half-positionality, but also to determinacy.

Theorem 4.2.1. *[Kop06] All concave and prefix-independent goals are determined and half-positional.*

4.2.2 Determining the winner

In the case the game represents a system-environment interaction, a strategy for the system constitutes a plan of scheduling of the activities in the states it controls. Hence, determining the winner of a game, i.e. determining which player has a winning strategy, is a fundamental problem. Half-positionality and determinacy allow us to reduce the problem on games to a problem on graphs through an exponential guess.

Suppose a goal W is half-positional for player 1, if player 1 has a winning strategy on game $G = (A, W)$, the he has also a winning memoryless one. Hence, we simply need to evaluate all the memoryless strategies of player 1, and if we do not find a winning one we conclude that player 0 is the winner. Since the memoryless strategies are functions τ from V_1 to E, they are finite in number, and they can be represented in polynomial space. Each memoryless strategy forms a subarena A' in the arena A, obtained from A by removing for all the nodes v controlled by player 1, all edges exiting from those nodes but the one used in the strategy, i.e. $\tau(v)$. Hence, for each node $v \in V_1$ in A' there exists only one exiting edge, in such situation player 1 has no choices. Actually the arena can be viewed as a graph whose nodes are all controlled by player 0. Upon such a graph we may apply an easier algorithm and check whether player 0 has a way to beat the strategy τ by finding a path with color sequence in W. Following this idea, we prove the next lemma.

Lemma 4.2.3. *Let W be a k-colored, half-positional and determined goal, such that for all initialized graphs $G = (V, E, v_{ini})$ there exists a polynomial time algorithm determining whether there exists an infinite path starting at v_{ini} with color sequence in W. Then, for each k-colored game $G = (A, W)$ the problem asking whether there exists winning strategy for player 1 is in NP and the problem asking whether there exists a winning strategy for player 0 is in $Co - NP$.*

Proof. By half-positionality, if player 1 has a winning strategy, he has a memoryless one. The number of memoryless strategies is finite and each one of them can be represented in polynomial space in the size of the problem. (It is the number of possible functions that associate a node in V_1 with an edge exiting from that node). So, in polynomial time we can guess a memoryless strategy τ, and verify that it is a winning strategy, using the following algorithm. We construct the subarena A', obtained from A by removing all the edges of player 1 that are not used by τ. It is easy to see the set of plays on A consistent with τ is equal to the set of all plays on A'. We have that τ is a winning strategy for player 1 in A if and only if all the plays on A' are winning for player 1. Observe that in A' player 1 actually has no choice. so we may consider that a play in A' is

constructed only by player 0. Thus, player 0 has a strategy that beats τ if and only if there exists a path starting from v_{ini}, with color sequence in W in the graph of A'. By hypothesis we known to solve the problem in polynomial time.

Since, we evaluate whether there exists a wining strategy for player 1, by guessing a polynomial certificate that can be checked in polynomial time, we obtain that the problem is in NP. Hence, the complementary problem asking whether there exists a winning strategy for player 0 is in $Co - NP$. □

4.3 Determining the winner

When a k-colored game represents the interaction between a scheduler and the environment, determining whether the system has a winning strategy, allows us to determine whether a scheduler can force a desired specification. In the case of the fairness and priority goals, we show that determining whether player 0 has winning strategy is $Co - NP$-complete and that determining whether player 1 has a winning strategy is NP-complete. We also propose an algorithm that allows us to determine the winner.

4.3.1 Membership

We first prove that the fairness and priority goals are determined and half-positional for player 1 by using Kopczyńsky's theorem. By using lemma 4.2.3 and the algorithms on graphs for determining fair and priority path discussed in Chapters 2 and 3, we obtain a NP algorithm that allows us to determine whether there exists a winning strategy for player 1.

The following lemmas prove that the complements of fairness and priority goals are prefix-independent and concave. Since a goal is prefix-independent if and only if its complement is prefix independent, for this property the following lemma is sufficient.

Lemma 4.3.1. *For every frequency vector $f \in \mathbb{Q}^k$, the priority goals W_{af}, W_{uf} and the fair goals W_{bn}, W_{bl} are prefix independent.*

Proof. Let $x \in [k]^\omega$, and let $y \in [k]^*$, we prove that for all the goals, the color sequence x is winning if and only if $y \cdot x$ is winning. We distinguish the following cases.

1. (*asymptotic frequency f*) x has asymptotic frequency f if and only if for each color $i \in [k]$ we have $\lim_{n\to+\infty} \frac{|x^{\leq n}|_i}{n} = f_i$ Since $|x^{\leq n}|_i = |(y \cdot x)^{\leq n+|y|}|_i - |y|_i$ for by setting $m = n + |y|$ we have $|(y \cdot x)^{\leq m}|_i = |x^{\leq m-|y|}| + |y|_i$. Hence x has asymptotic frequency f, if and only if for each color $i \in [k]$ we have $\lim_{m\to+\infty} \frac{|(y\cdot x)^{\leq m}|_i}{m} = \lim_{n\to+\infty} \frac{|x^{\leq n}|_i - |y_i|}{n-|y|} = f_i$ and this happens if and only if $y \cdot x$ has asymptotic frequency f.

2. (*uniform frequency f*) We already proved that x has asymptotic frequency f if and only if $y \cdot x$ does, it remains to prove that such a frequency is uniform on x if and only if it is uniform on $y \cdot x$.

   [only if] Since x has asymptotic frequency f, there exists a constant C such that for all $n \in \mathbb{N}$ and $i \in [k]$ we have $-C \leq n \cdot f_i - |x^{\leq n}|_i \leq C$. Let $m = n + |y|$, and $C_i = |y| \cdot f_i - |y|_i$, let

$C' = \max C_i \mid i \in [k]$. Let $C'' = \max\{|m \cdot f_i| - |y^{leqm}|_i \mid i \in [k], m \leq |y|\}$. Then, we have for all $m \in \mathbb{N}$, $\min\{-C - C', -C''\} \leq m \cdot f_i - |(y \cdot x)^{\leq m}|_i \leq \max\{C + C', C''\}$. Hence, $y \cdot x$ has uniform frequency f.

[if] Since $y \cdot x$ has asymptotic frequency f, there exists a constant C such that if for all $m \in \mathbb{N}$ and $i \in [k]$ we have $-C \leq m \cdot f_i - |(y \cdot x)^{\leq m}|_i \leq C$. Let $m = n + |y|$, and $C_i = |y| \cdot f_i - |y|_i$. Then, we have for all $n \in \mathbb{N}$, $-C - C' \leq n \cdot f_i - |x^{\leq n}|_i \leq C + C'$. Hence, x has uniform frequency f.

3. (*Bounded and Balanced*) Since the balance (resp., bounded in difference)property is equivalent to the asymptotic (resp., uniform) frequency-f property with f_i equal to $\frac{1}{k}$ for all colors $i \in [k]$, the thesis holds.

□

Lemma 4.3.2. *For every frequency vector $f \in \mathbb{Q}^k$, the priority goals W_{af}, W_{uf} and the fair goals W_{bn}, W_{bl} are convex.*

Proof. Let $y, z \in [k]^\omega$ and $x \in y \otimes z$. We prove that if y and z are winning, then so is x. We have that $x = x_1 \dots x_i \dots$ where $y = x_1 x_3 \dots x_{2k+1} \dots$ and $z = x_2 x_4 \dots x_{2k} \dots$. Also, for all $n \in \mathbb{N}$ there are two indexes n_y, n_z such that $n = n_y + n_z$ and $\overline{N_{um}}(x^{\leq n}) = \overline{N_{um}}(y^{\leq n_y}) + \overline{N_{um}}(z^{\leq n_z})$, for all $a, b \in [k]$. We distinguish the following cases.

1. (*asymptotic frequency-f*) Given that y and z have asymptotic frequency f, we have that, for all $a \in [k]$ and for all $\varepsilon > 0$, there exists $h(\varepsilon) > 0$ such that for all $n > h(\varepsilon)$, it holds that $\left|\frac{|y^{\leq n}|_a}{n} - f_a\right| \leq \varepsilon$ and $\left|\frac{|z^{\leq n}|_a}{n} - f_a\right| \leq \varepsilon$. Hence, given $\varepsilon > 0$, let $n > 0$ be such that $n_y \geq h(\frac{\varepsilon}{2})$ and $n_z \geq h(\frac{\varepsilon}{2})$. Such n exists, due to the definition of the shuffle operation. For all $n' > n$ we have that:

$$\begin{aligned}\left|\frac{|x^{\leq n'}|_a}{n'} - f_a\right| &= \left|\frac{|y^{\leq n'_y}|_a + |z^{\leq n'_z}|_a - (n'_y + n'_z) f_a}{n'_y + n'_z}\right| \\ &\leq \left|\frac{|y^{\leq n'_y}|_a - n'_y \cdot f_a}{n'_y + n'_z}\right| + \left|\frac{|z^{\leq n'_z}|_a - n'_z \cdot f_a}{n'_y + n'_z}\right| \\ &\leq \left|\frac{|y^{\leq n'_y}|_a}{n'_y} - f_a\right| + \left|\frac{|z^{\leq n'_z}|_a}{n'_z} - f_a\right| \leq \varepsilon.\end{aligned}$$

So, the color sequence x has asymptotic frequency vector f.

2. (*uniform frequency-f*) Since y and z have uniform frequency f, then x has asymptotic frequency f by the previous point. Moreover, there exist two constants $C_y, C_z \in \mathbb{N}$ such that for all $i \in [k]$ and for all $n > 0$, $|n \cdot f_i - |y^{\leq n}|_i| < C_y$ and $|n \cdot f_i - |z^{\leq n}|_i| < C_z$. Therefore, let $C_x = C_y + C_z$, for all $i \in [k]$ and $n \in \mathbb{N}$ we have $|n \cdot f_i - |x^{\leq n}|_i| \leq |n_y \cdot f_i - |y^{\leq n_y}|_i| + |n_z \cdot f_i - |z^{\leq n_z}|_i| \leq C_x$. Hence, the sequence x has uniform frequency f.

3. (*balanced and bounded*) Since the balance (resp., bounded in difference)property is equivalent to the asymptotic (resp., uniform) frequency-f property with f_i equal to $\frac{1}{/}k$ for all colors $i \in [k]$, the thesis holds.

□

By Kopczyński theorem, and the previous lemmas, the following corollary follows.

Corollary 4.3.1. *For every frequency vector $f \in \mathbb{Q}^k$, the priority goals W_{af}, W_{uf} and the fair goals W_{bn}, W_{bl} are determined and half-positional for player* 1.

Recall that from Section 3.5, we have a polynomial algorithm that allows us to determine a fair or priority infinite path reachable from a starting state in a k-colored graph. Let ρ be such a infinite path and let ρ' be the path connecting v_{ini} to the starting node of ρ. Then, by prefix independence the new infinite path $\rho' \cdot \rho$ has the same asymptotic (resp., uniform) frequency of ρ. Since constructing a path from v_{ini} to the starting node of ρ' is polynomial in the size of the game graph, we have polynomial algorithms that allow us to compute a fair or priority path starting from a the initial node v_{ini} in any k-colored graph. By Lemma 4.2.3, we are able to construct an NP algorithm determining whether there exists a winning strategy for player 1.

Corollary 4.3.2. *Given a k-colored game with balanced, bounded, asymptotic frequency-f, or uniform frequency-f goal, the problem asking whether there exists a winning strategy for player* 1 *is in NP, the problem asking whether there exists a winning strategy for player* 0 *is in $Co - NP$.*

4.3.2 Hardness

In this section we prove that determining whether there exists a winning strategy for player 0 is an $Co - NP$-hard problem, by reducing it to the satisfiability problem of a boolean formula, which is a known $Co - NP$ hard problem. Hence, determining whether there exists a winning strategy for player 1 is a NP-hard problem.

Lemma 4.3.3. *Given a boolean formula ψ in conjunctive normal form, there exists a k-colored arena A such that the following are equivalent (i) ψ is a tautology, (ii) there exists a winning strategy for player* 0 *in the game $G = (A, W_{bl})$, and (iii) there exists a winning strategy for player* 0 *in the game $G = (A, W_{bn})$.*

Proof. Let n be the number of clauses of ψ and m be the number of its variables, then we can write $\psi = \wedge_{i=1}^{n} \psi_i$, where each ψ_i is a disjunction of literals. In the following we define $\psi(x)$ as the set of all clauses in which x appears in positive form, and $\psi(\overline{x})$ as the set of all clauses in which x appears negated.

We construct the following $(n+1)$-colored arena $A = (V_0, V_1, v_{\text{ini}}, E)$, where the set of colors corresponds to the set of clauses of ψ with the added control color $n+1$. The description of the arena A makes use of *uncolored edges*, i.e., edges not labeled by any color. Clearly, such an edge can be represented in our framework by a sequence of $n+1$ edges, each labeled by a different color. The arena A is composed by m subarenas A_j, one for each variable x_j. Every subarena A_j has a starting node v_j, an ending node v'_j and two sequences of nodes: $\{v_{j,i}\}_{i=1}^{n}$, $\{\overline{v}_{j,i}\}_{i=1}^{n}$ where

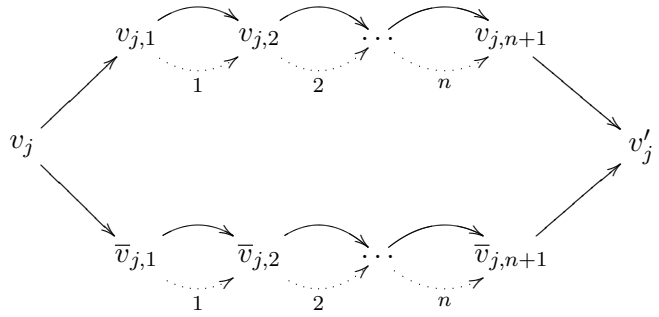


Figure 4.1: The j-th subgraph A_j of A. The dotted edge from $v_{j,i}$ to $v_{j,i+1}$ is present if and only if $\psi_i \in \psi(x_j)$, and analogously for the lower branch.

every node is associated with a clause. There is an uncolored edge from v_j to $v_{j,1}$ and from v_j to $\overline{v}_{j,1}$. Moreover, if we define $v_{j,n+1} = \overline{v}_{j,n+1} = v'_j$, we have that for all $1 \leq i \leq n$, ***(i)*** there is an uncolored edge from $v_{j,i}$ to $v_{j,i+1}$ and from $\overline{v}_{j,i}$ to $\overline{v}_{j,i+1}$, ***(ii)*** if $\psi_i \in \psi(x)$ then there is an i-colored edge from $v_{j,i}$ to $v_{j,i+1}$, and ***(iii)*** if $\psi_i \in \psi(\overline{x})$ then there is an i-colored edge from $\overline{v}_{j,i}$ to $\overline{v}_{j,i+1}$. We call the sequence $\{v_{j,i}\}_i$ the *upper branch* of A_j and the sequence $\{\overline{v}_{j,i}\}_i$ the *lower branch* of A_j. The arena A is constructed by connecting the subarenas A_j as follows: for all $1 \leq j \leq m-1$ there is an uncolored edge from v'_j to v_{j+1} and an $n+1$-colored edge from v'_m to v_1.

The construction of A is concluded by partitioning the set of nodes in $V_1 = \{v_1, \ldots, v_m\}$ and $V_0 = V - V_1$. Intuitively, every subarena A_j represents a truth choice for the variable x_j. This choice is made by player 1 with the aim to skip the passage through some clauses. On the other hand, as soon as there is the chance, player 0 tries to pass through each clause once during a single loop, in order to balance the clauses' colors with the control color $n+1$. Let $G = (A, W_{bl})$ and $G' = (A, W_{bn})$, we now show the correctness of the above construction. In the following, we write $\widetilde{v}_{j,i}$ to mean either $v_{j,i}$ or $\overline{v}_{j,i}$.

[If] If ψ is a tautology, then the winning strategy for player 0 in both games G and G' may be summarized as follows: as soon as there is a chance, pass through an edge of color ψ_i; then, do not pass through such an edge again, until we pass again through v_1. Formally, the strategy of player 0 is the following: each time the play is in a node $\widetilde{v}_{j,i}$, player 0 chooses to reach $\widetilde{v}_{j,i+1}$ through the ψ_i-colored edge if and only if ψ_i does not appear in the least suffix of the partial play starting with v_1. then player 1 does not have a choice and it follows the only possible outgoing edge. We observe that during a single loop from v_1 to itself, a strategy of player 1 is a truth-assignment to the variables of ψ: precisely for every subarena A_j, player 1 chooses to follow the upper branch if and only if x_j is true. Since ψ is a tautology, any such assignment is a satisfiable assignment, i.e., given such an assignment $a : \{x_1, \ldots, x_n\} \to \{T, F\}$, for each clause ψ_i, there exists a variable x such that ψ_i is true also due to the value $a(x)$. This means that player 0 can pass through a ψ_i-colored edge at least once during a single loop, and thanks to his strategy, he will pass through such an edge exactly once. Thus, during each loop, the uncolored edges are already perfectly balanced, and

the edges added by player 0 are balanced thanks to the last $n+1$-colored edge. Thus, during the infinite play, the color differences are always zero when the play is in node v_1. Since the loops from v_1 to itself have bounded length, the color differences are bounded during the play. Thus every infinite play consistent with the strategy is bounded and it is balanced too, because in [BFMM09] we proved that a bounded path is balanced too.

[Only If]. If ψ is not a tautology, then there is a memoryless winning strategy for player 1 on G and on G': player 1 follows a truth assignment of the variables of ψ that does not satisfy ψ. For such an assignment there is an unsatisfiable clause ψ_i. So, during a loop from v_1 to itself, if player 1 follows this strategy, player 0 cannot pass through any ψ_i-colored edge. Thus, at the end of the loop the color difference between color ψ_i and color $n+1$ is increased by one. Every play ρ is an infinite concatenation of simple loops from v_1 to itself. Since those loops have maximum length $l \leq |E|$, for all $j \in \mathbb{N}$ we have $d_{iff\,i,n+1}(\rho^{\leq j}) \geq \frac{j}{l}$, and thus $\lim_{j\to+\infty} \frac{d_{iff\,i,n+1}(\rho^{\leq j})}{j} \geq \frac{1}{l}$. This means that every play consistent with said strategy of player 1 is not balanced, and hence not bounded. □

Theorem 4.3.1. *Given a k-colored game G with balanced (resp., bounded, frequency-f) goal, the problem asking whether there exists a winning strategy for player 0 is Co-NP-complete.*

Proof. By Lemma 4.2.3 and Lemma 4.3.3, we have that the problems for the balance and the bounded goal are Co-NP-complete. Since the bounded goal is a special case of frequency-f goal (for $f_i = 1/k$), we have that the frequency-f problem is Co-NP-hard too. Since by Lemma 4.2.3 the problem for frequency-f is in Co-NP, it is Co-NP-complete. □

This Co-NP-completeness result may be regarded as essentially negative. In fact, the algorithm showing membership in NP, once converted into a deterministic form, simply suggests to try each one of the (exponential) memoryless strategies of player 1 in the game, and solve a linear program to determine whether it is winning.

4.4 Computing the winning strategy

In the previous section we discussed an NP algorithm determining whether there exists a winning strategy for player 1. Such an algorithm uses memoryless strategies as certificates, hence, when it terminates with a valid certificate, it also provides a winning strategy for player 1. However, when there does not exist any winning strategy for player 1, the algorithm terminates without giving any information about the winning strategy of player 0. Such a strategy is quite interesting, since it says how a scheduler should behave in order to ensure the construction of a fair or priority path regardless of the environment's choices. In this section, we show how to construct a winning strategy for player 0 in memory-update form, by induction on the number of edges exiting from nodes belonging to player 1. We start from the arenas controlled by player 0 where there is only one edge exiting from nodes of player 1, on such arenas the winning strategy is given by a fair or priority path computable through the technique of Chapters 2 and 3. We decompose a complex arena $A = (V_0, V_1, E)$ in two subarenas $A_i = (V_0, V_1, E_i)$ with the same set of nodes but a reduced set of edges exiting from node of player 1. Using the winning strategies computed by

induction on the subarenas, we are able to compose them and form a winning strategy for the complex arena.

At every induction step we propagate the following inductive hypothesis on each arena $A = (V_0, V_1, E)$.

1. We computed a strategy $\tau = (M, \psi, \tau_\mu, \mu_0)$ in memory update form and the winning set $D \subseteq V_0 \cup V_1$ for τ.
2. We know that D is the greatest possible wining set for a strategy of player 0 on A, i.e., for each $u \notin D$, there exists a winning strategy for player 1 on A at u.

Hence, when the algorithm halts, we completely determine the set D of all and only the nodes from which there exists a winning strategy for player 0. Moreover, we also have a strategy in memory-update form that allows player 0 to win from every point in D.

As preliminaries to the algorithm we discuss a property of strategies for prefix-independent goals, that allows us to state that nodes not belonging to the maximal winning set D are not reachable by nodes in D with a path consistent with τ.

Lemma 4.4.1. *Let $A = (V_0, V_1, E)$ be a k-colored arena, W be a prefix-independent goal, τ be a strategy of player 0 on A, and D be the winning set of τ. Let $u \notin D$ be a node such that there exists a winning strategy τ' for player 1. Then, for all nodes $v \in D$ there does not exists a path from v to u consistent with τ.*

Proof. Suppose by contradiction that there exists a path π from v to u consistent with τ, then we can construct a strategy τ_1 for player 1 winning at v against τ. Indeed, let $\tau_1(v) = \pi(1)$; for all $n < |\pi|$ let $\tau_1(\pi^{\leq n}) = \pi(n+1)$; for all $\pi' \in E^*$ starting at u let $\tau_1(\pi \cdot \pi') = \tau'(\pi')$. The play consistent with τ and τ_1 forces the path π at the beginning and it has the form $\pi \cdot \rho$ where ρ is an infinite path starting at u consistent with τ'. Hence, ρ is losing for player 0, by prefix independence $\pi \cdot \rho$ is losing and τ is not a winning strategy. This contradicts the hypothesis. □

4.4.1 Base Step

As a base case, consider a game arena A such that from all the nodes controlled by player 1, there is only one exiting edge. Such an arena can be considered as a game graph where all nodes are controlled by player 0. Hence, we can compute a winning strategy for player 0 starting at a node v as a fair or priority path through the polynomial algorithm discussed in Chapters 2 and 3. However, we need to construct a winning strategy for the maximal winning set of A. Hence, we iteratively construct the set R of nodes from which there is no fair or priority path and a sequence of pairs $\{(M_i, \psi_i, \tau_{\mu,i}, \mu_{0,i}), R_i)\}_{i=1}^{l}$ memory-update-form strategy and winning set for that strategy such that $\cup_{i=1}^{l} R_i \cup R = V_0 \cup V_1$. The construction is by induction on the size of the set R' of nodes v such that we still do not whether there exists a winning path from v. At each step we pick a node $v \in R'$, and we determine whether there exists a fair or priority path in R' through the appropriate polynomial algorithm. If such a path does not exist v is put in the losing set R and removed from R'. If the path does exist, then we construct a strategy τ_i winning on the set R_i of nodes belonging to the found path. Such nodes are then removed from R'. We show in the next subsection, how the strategy τ_i is constructed by case analysis on the type of goal.

Once the sequence $\{(M_i, \psi_i, \tau_{\mu,i}, \mu_{0,i}), R_i)\}_{i=1}^{l}$ is constructed, we can construct a strategy $\tau = (M, \psi, \tau_\mu, \mu_0)$ winning on $D = \cup_{i=1}^{l} R_i$ as follows.

1. $M = \cup_{i=1}^{l}(\{i\} \times M_i)$
2. For all $(i, m, e) \in M \times E$ we have $\psi(i, m, e) = \psi_i(m, e)$.
3. For all $(i, m) \in M$ we have $\tau_\mu(i, m) = \tau_{\mu,i}(m)$
4. For all $u \in R_i$ we have $\mu_0(u) = \mu_{0,i}(u)$.

The strategy is winning on D because for each $u \in R_i$ it behaves like the strategy τ_i winning on u. Moreover, since $D \cup R = V_0 \cup V_1$ and R is a set of losing nodes we have that D is the maximal winning set on A. In the following, we show how to compute for a node v, from which a fair or priority path ρ starts, a strategy τ winning on the set of nodes R belonging to ρ.

Uniform Frequency

For the uniform frequency-f goals (hence, also for the bounded difference ones) if the polynomial algorithm finds a frequency-f path, it computes an overlapping set of simple loops $\sigma_1, \ldots, \sigma_l$ reachable from a node v and having a linear combination of ratio f. Such set of loops can be composed in polynomial time to form a perfectly balanced loop σ reachable from v. Let π be a path from v to σ, then $\rho = \pi \cdot (\sigma)^\omega$ is a path starting from v with uniform frequency f.

Hence, $\tau = (M, \psi, \tau_\mu, \mu_0)$ just needs to remember the part of π or the part of σ constructed so far.

1. We use as a memory state M the set of all prefixes of π and σ. Hence, $M = \{\pi^{\leq n} \mid 0 \leq n \leq |\pi|\} \cup \{\sigma^{\leq n} \mid 0 < n \leq |\sigma|\}$.
2. The update function $\psi : M \times E \to M$ just updates the prefixes, hence, it is such that
   (a) for all $n < |\pi|$, $\psi(\pi^{\leq n}, \pi(n+1)) = \pi^{\leq n+1}$,
   (b) $\psi(\pi, \sigma(1)) = \sigma^{\leq 1}$,
   (c) for all $n < |\sigma|$, $\psi(\sigma^{\leq n}, \sigma(n+1)) = \sigma^{\leq n+1}$,
   (d) $\psi(\sigma, \sigma(1)) = \sigma^{\leq 1}$.
3. The choice function $\tau_\mu : M \times E$ is such that
   (a) for all $n < |\pi|$, $\tau_\mu(\pi^{\leq n}) = \pi(n+1)$,
   (b) $\tau_\mu(\pi) = \sigma(1)$,
   (c) for all $n < |\sigma|$, $\tau_\mu(\sigma^{\leq n}) = \sigma(n+1)$,
   (d) $\tau_\mu(\sigma) = \sigma(1)$.
4. For the starting memory we only need to set $\mu_0(v) = \varepsilon$.

5. Since, player 1 has no choice, following the strategy written so far, player 0 will surely follow the path ρ, hence, for all other paths and memory values the value of U and τ_μ is not important, and can be set arbitrary.

The above strategy is surely winning on A at v. However, one can notice that, by prefix independence, it should win also on the set R of nodes belonging to the path ρ, because the suffix after a node in R is also winning. In order to allow the strategy to be winning also at such nodes u, it is sufficient that the starting memory on u is the memory developed by τ on the path starting from v till the first occurrence of u. Precisely, for all $u \in R \setminus \{v\}$, we set $\mu_0(u) = \psi^*(m_0, \pi_u)$ where π_u is the shortest prefix of ρ ending with u. If we write $\rho = \pi_u \cdot \rho_u$ it is easy to see that ρ_u is the only path starting at u and consistent with τ, and by prefix independence it is a winning path.

Asymptotic Frequency

For the asymptotic frequency-f goals (hence, also for the balanced ones) the polynomial algorithm computes a connected set $\{\sigma_1, \ldots, \sigma_l\}$ of simple loops reachable from and v and having a linear combination with coefficients $c_1, \ldots c_l$ of ratio f. Let π_i be a path connecting σ_i to $\sigma_{i+1 m_{od} l}$, then the loops, the coefficients and the paths constitute a polynomial representation of the path $\rho' = \prod_{i=1}^{+\infty} (\sigma_i)^{i \times c_i} \pi_i$. Such a path has asymptotic frequency f and is reachable from v. By choosing π as path from v to σ_1, we have that $\rho = \pi \cdot \rho'$ is a path starting from v with asymptotic frequency f.

Hence, $\tau = (M, \psi, \tau_\mu, \mu_0)$ needs to remember the part of π or the indexes i, j, the part of σ_j or π_j constructed so far, and in the case of σ_j the number of times c the loop has been created so far.

1. We use as a memory state M the set of all prefixes of π, and the set of prefixes of σ_j and π_j for all $j \in [l]$ and $i \in \mathbb{N}$. Hence, $M = \{\pi^{\leq n} \mid 0 \leq n \leq |\pi|\} \cup \mathbb{N} \times [l] \times \{\pi_j^{\leq n} \mid 0 < n \leq |\pi_j|\} \cup \mathbb{N} \times [l] \times \mathbb{N} \times \{\sigma_j^{\leq n} \mid 0 < n \leq |\sigma_j|\}$. A memory $(i, j, \pi_j^{\leq n})$ means the play is at the i-th step of the outer product, at the j-th step of the inner product and the play is constructing π_j. A memory $(i, j, c, \sigma_j^{\leq n})$ means the play is at the i-th step of the outer product, at the j-th step of the inner product and the play is constructing σ_j after constructing it already c times.

2. The update function $\psi : M \times E \to M$ just updates the prefixes, hence, it is such that

   (a) for all $n < |\pi|$, $\psi(\pi^{\leq n}, \pi(n+1)) = \pi^{\leq n+1}$,

   (b) $\psi(\pi, \sigma_1(1)) = (1, 1, 0, \sigma_1^{\leq 1})$

   (c) for all i, j, c, n with $n < |\sigma_j|$, $\psi(i, j, c\sigma_j^{\leq n}, \sigma_j(n+1)) = (i, j, c, \sigma_j^{\leq n+1})$,

   (d) for all i, j, c with $c < i \times c_j$, $\psi(i, j, c, \sigma_j) = (i, j, c+1, \sigma_j^{\leq 1})$.

   (e) for all i, j, $\psi(i, j, i \cdot c_j, \sigma_j) = (i, j, \pi_j^{\leq 1})$

   (f) for all i, j, n with $n < |\pi_j|$, $\psi(i, j, \pi_j^{\leq n}) = (i, j, \pi_j^{\leq n+1})$

   (g) for all i, j with $j < l$, $\psi(i, j, \pi_j) = (i, j+1, 0, \sigma_{j+1}^{\leq 1})$

   (h) for all i, $\psi(i, l, \pi_l) = (i+1, 1, 0, \sigma_1^{\leq 1})$.

3. The choice function $\tau_\mu : M \times E$ is such that
   (a) for all $n < |\pi|$, $\tau_\mu(\pi^{\leq n}) = \pi(n+1)$,
   (b) $\tau_\mu(\pi) = \sigma_1(1)$
   (c) for all i, j, c, n with $n < |\sigma_j|$, $\tau_\mu(i, j, c\sigma_j^{\leq n}, \sigma_j(n+1)) = \sigma_j(n+1)$,
   (d) for all i, j, c with $c < i \times c_j$, $\tau_\mu(i, j, c, \sigma_j) = \sigma_j(1)$.
   (e) for all i, j, $\tau_\mu(i, j, i \cdot c_j, \sigma_j) = \pi_j(1)$
   (f) for all i, j, n with $n < |\pi_j|$, $\tau_\mu(i, j, \pi_j^{\leq n}) = \pi_j(n+1)$
   (g) for all i, j with $j < l$, $\tau_\mu(i, j, \pi_j) = \sigma_{j+1}(1)$
   (h) for all i, $\tau_\mu(i, l, \pi_l) = \sigma_1(1)$.
4. The starting memory is $\mu_0(v) = \varepsilon$.
5. Since, player 1 has no choice, following the strategy written so far, player 0 will surely follow the path ρ, hence, for all other paths and memory values the value of U and τ_μ is not important, and can be set arbitrary.

The above strategy is surely winning on A at v. However, one can notice that, by prefix independence, it should win also on the set R of nodes belonging to the path ρ, because the suffix after a node in R is also winning. In order to allow the strategy to be winning also at such nodes u, it is sufficient that the starting memory on u is the memory developed by τ on the path starting from v till the first occurrence of u. Precisely, for all $u \in R \setminus \{v\}$, we set $\mu_0(u) = \psi^*(m_0, \pi_u)$ where π_u is the shortest prefix of ρ ending with u. If we write $\rho = \pi_u \cdot \rho_u$ it is easy to see that ρ_u is the only path starting at u and consistent with τ, and by prefix independence it is a winning path.

Approximated Asymptotic Frequency

For the asymptotic frequency f the need for an infinite memory, does not allow a scheduler to be able to update the memory forever. Hence, for practical purposes, we cannot use the asymptotic frequency-f path found. However, the path $\rho' = \pi \prod_{i=1}^{+\infty} \prod_{j=1}^{l} (\sigma_j)^{\leq d \cdot c_j} \pi_j$ may constitute a reasonable approximation of a path with frequency f for a great enough fixed value of d. By setting $\sigma = \prod_{j=1}^{l} (\sigma_j)^{\leq d \cdot c_j} \pi_j$, we have that $\rho' = \pi \cdot (\sigma)^\omega$, hence we can construct a strategy following ρ' like discussed above for the paths with uniform frequency.

Observe that ρ' has uniform frequency f', which is given by the frequency of the loop σ'_i given by $\prod_{j=1}^{l} (\sigma_j)^{\leq d \cdot c_j} \pi_j$. By construction of ρ and ρ' we know that for each color $i \in [k]$ it holds that $f_i = \frac{\sum_{j=1}^{l} c_j |\sigma_j|_i}{\sum_{j=1}^{l} c_j |\sigma_j}$, and $f'_i = \frac{d \cdot \sum_{j=1}^{l} (c_j \cdot |\sigma_j|_i) \sum_{j=1}^{l} |\pi_j|_i}{d \cdot \sum_{j=1}^{l} (c_j \cdot |\sigma_j|) \sum_{j=1}^{l} |\pi_j|}$. Since $\sum_{j=1}^{l} |\pi_j|$ is finite, there exist constants d_i such that $|\sum_{j=1}^{l} |\pi_j|_i| \leq d_i \cdot |\sum_{j=1}^{l} (c_j \cdot |\sigma_j|_i)|$ and a constant d' such that $|\sum_{j=1}^{l} |\pi_j|| \leq d \cdot |\sum_{j=1}^{l} (c_j \cdot |\sigma_j|)|$. Hence, we obtain that for each $i \in \mathbb{N}$ it holds that $\frac{d-d_i}{d+d'} f_i \leq f'_i \leq \frac{d+d_i}{d-d'} f_i$, thus the greater is d the closer f'_i is to the desired frequency f_i.

4.4.2 Shuffle Strategy

For the inductive case, we consider an initialized arena $A = (V_0, V_1, E)$ such that there are $n > |V_1|$ edges exiting from nodes controlled by player 1. In order to reduce the computation of a winning strategy on A, to a computation on less complex arenas, we decompose A in two subarenas. In this subsection we show how to combine winning strategies on the two subarenas in order to obtain winning strategies on A. We complete the construction on the inductive step of the algorithm in the next subsection.

Precisely, consider a node $v \in V_1$ from which there are at least two exiting edges. Then, we can decompose A in two subarenas equal to A but for the fact that they contain only a part of the set of edges exiting from v. Formally, we can partition the set $v^{\rightarrow}$ of edges exiting from v in two sets E_α, E_β. Then, for all $i \in \{\alpha, \beta\}$, we construct the subarena $A_i = (V_0, V_1, E'_i,)$ obtained from A by removing all edges exiting from v but the edges in E_i, i.e, $E'_i = E \setminus (v^{\rightarrow} \setminus E_i)$.

Definition 4.4.1. v-partition

Let $A = (V_0, V_1, E)$ be a k-colored arena, $v \in V_1$ a node from which there are at least two outgoing edges, and E_α, E_β a partition of the set of edges $v^{\rightarrow}$ exiting from v in two non-empty, disjoint sets. Then a v-partition of A with respect to E_α, E_β is the pair of subarenas A_α, A_β such that $A_i = (V_0, V_1, E'_i)$ where $E'_i = E \setminus (v^{\rightarrow} \setminus E_i)$.

Consider two strategies τ_α and τ_β for player 0 on A. We can construct a strategy τ that behaves like τ_α as long as the path does not pass through v. When the path passes through v, depending on what set E_i, the last edge exiting from v belongs to, τ behaves like τ_i when such strategy ignores all the paths developed in the subarena different than A_i. Precisely, if the last edge exiting from v belongs to E_α (resp., E_β), τ behaves like the strategy τ_α (resp., τ_β) when it ignores all the loops from v to v that start with an edge in E_β (resp., it ignores the starting path from the initial node to v and all the loops from v to v that start with an edge in E_α).

Before giving the formal definition of shuffle strategy, observe that, for all finite paths π starting at u in A, we can write π as a concatenation between a path π_u from u to v, a set of loops π_j from v to v, and a path π_v, that starts from v and never passes again through v. Hence, we can write $\pi = \pi_u(\prod_{j=1}^{l} \pi_j)\pi_u$. Observe that π_u is empty if $u = v$, the path π_u is empty only if π ends with v, and $\pi = \pi_u$ if π does not pass through v.

Definition 4.4.2. Shuffle Strategy

Let (A_α, A_β) be a v-partition of an arena $A = (V_0, V_1, E)$ with respect to E_α, E_β. Let $u \in V_0 \cup V_1$, τ_α a strategy of player 0 on A_α and τ_β a strategy of player 0 on A_β.

Then the shuffle strategy $\tau = \tau_\alpha \otimes_v \tau_\beta$ is a strategy of player 0 on the arena A such that the value $\tau(\pi)$ (resp., $\tau(u)$) for every partial play π (resp., node u) is constructed as follows.

1. *Suppose $u \neq v$ then $\tau(u) = \tau_\alpha(u)$.*
2. *Suppose π does not pass through v, we have $\tau(\pi) = \tau_\alpha(\pi)$.*
3. *Suppose $\pi = \pi_u(\prod_{j=1}^{l} \pi_j)\pi_v$ passes at least once through v, and the first edge of π_v belongs to E_α. We can compute the subsequence of paths $\pi_{i_1}, \ldots, \pi_{i_{l'}}$ of all and only the paths $\pi_1, \ldots, \pi_l$ starting with edges in E_α. We have $\tau(\pi) = \tau_\alpha(\pi_u \cdot (\prod_{j=1}^{l'} \pi_{i_j}) \cdot \pi_v)$.*

4. *Suppose $\pi = \pi_u(\prod_{j=1}^{l} \pi_j)\pi_v$ passes at least once through v, and the first edge of π_v belongs to E_β. We can compute the subsequence of paths $\pi_{i_1}, \ldots, \pi_{i_{l'}}$ of all and only the paths $\pi_1, \ldots, \pi_l$ starting with edges in E_β. We have $\tau(\pi) = \tau'_\beta((\prod_{j=1}^{l'} \pi_{i_j}) \cdot \pi_v)$.*

5. *Suppose $\pi = \pi_u(\prod_{j=1}^{l} \pi_j)\pi_v$ passes at least once through v, and π_v is empty. Then, we do not need to set τ on such a path, since it ends with v which is a node belonging to player* 1.

The elements in $(V_0 \cup V_1) \cup E^+$ for which the shuffle τ has not been specified, in general, are meaningless for player 0 (because they are path ending in nodes of V_1), and, hence can be set arbitrary.

The memory-update form of $\tau = \tau_\alpha \otimes_v \tau_\beta$ can be computed easily starting from the memory update forms of $\tau_\alpha = (M_\alpha, \psi_\alpha, \tau_{\mu,\alpha}, \mu_{0,\alpha})$ and $\tau_\beta = (M_\beta, \psi_\beta, \tau_{\mu,\beta}, \mu_{0,\beta})$. Indeed, τ behaves like either τ_α or τ_β depending on the last edge from v. Furthermore, when τ behaves like τ_α (resp., τ_β) it keeps into account only the part of play developed while behaving like τ_α (resp., τ_β). Hence, τ is actually using both the memories from τ_α and τ_β and switches from one to the other, every time it passes through v and the arena is switched. Hence, the memory set for τ is $M = \{\alpha, \beta\} \times M_\alpha \times M_\beta$. A memory value $m = (\alpha, m_\alpha, m_\beta)$ (resp., $m = (\beta, m_\alpha, m_\beta)$) means that τ is behaving like τ_α (resp., τ_β) with memory m_α (resp., m_β), moreover only the memory m_α (resp., m_β) is updated, while the memory m_β (resp., m_α) is just stored for the next time player 1 decides to choose and edge in E_β (resp., E_α). Hence, the memory-update form $\tau = (M, \psi, \tau_\mu, \mu_0)$ is the following.

Definition 4.4.3. Memory-update form of the shuffle Strategy

1. $M = \{\alpha, \beta\} \times M_\alpha \times M_\beta$.

2. $\psi : M \times E \to M$ *is such that:* (i) *for all* $(i, m_\alpha, m_\beta, e) \in M \times (E \setminus v^\rightarrow)$ *we have* $\psi(\alpha, m_\alpha, m_\beta, e) = (\alpha, \psi_\alpha(m_\alpha, e), m_\beta)$ *and* $\psi(\beta, m_\alpha, m_\beta, e) = (\beta, m_\alpha, \psi(m_\beta, e))$*;* (ii) *for all* $(i, m_\alpha, m_\beta, e) \in M \times E_j$ *we have* $\psi(\alpha, m_\alpha, m_\beta, e) == (j, \psi_\alpha(m_\alpha, e), m_\beta)$ *and* $\psi(\beta, m_\alpha, m_\beta, e) = (j, m_\alpha, \psi(m_\beta, e))$.

3. $\tau_\mu : M \to E$ *is such that for all* $(i, m_\alpha, m_\beta) \in m$ *we have* $\tau_\mu(i, m_\alpha, m_\beta) = \tau_{\mu,i}(m_i)$

4. *The starting memory function is* $\mu_0 : (V_0 \cup V_1) \to E$ *such that for all* $u \in V_0 \cup V_1$ $\mu_0(u) = (\alpha, \mu_{0,\alpha}(u), \mu_{0,\beta}(v))$.

Observe that by construction of the shuffle in memory update form we have that $\tau' = \tau_\beta \otimes_v \tau_\alpha = (M', \psi', \tau'_\mu, \mu'_0)$ is such that:

1. $M' = \{\alpha, \beta\} \times M_\beta \times M_\alpha$
2. for all $(i, m_\beta, m_\alpha, e) \in M \times E$ $\psi'(i, m_\beta, m_\alpha, e) = \psi(i, m_\alpha, m_\beta, e)$
3. for all $(i, m_\beta, m_\alpha) \in M$ $\tau'_\mu(i, m_\beta, m_\alpha) = \tau(i, m_\alpha, m_\beta)$
4. for all $u \in V_0 \cup V_1$ $\mu'_0(u) = (\beta, \mu_{0,\beta}(u), \mu_{0,\alpha}(v))$.

Hence, by inverting the order of memories from α and β in τ' we have that $\tau' = (M, u, \tau, m'')$ where $\mu''_0(u) = (\beta, \mu_{0,\alpha}(v), \mu_{0,\beta}(v))$. We call this property *almost commutativity* of the shuffle operation.

The following lemma shows that, for prefix-independent and convex goals, the shuffle strategy of two winning strategies is winning as well.

Lemma 4.4.2. *Let (A_α, A_β) be a v-partition of an arena $A = (V_0, V_1, E)$ with respect to E_α, E_β, and W be a prefix-independent and convex goal. Let $u \in V_0 \cup V_1$, τ_α a winning strategy of player 0 on A_α at u and τ_β a winning strategy of player 0 on A_β at v.*

Then the shuffle strategy $\tau = \tau_\alpha \otimes_v \tau_\beta$ is a winning strategy for player 0 on A at u.

Proof. We prove that the strategy τ is winning, showing that all plays ρ consistent with τ are winning by a case analysis on the type of play ρ.

1. Suppose ρ does not pass through v. Then, it is a path in A_α consistent with τ_α and, hence, is a winning path.

2. Suppose the path ρ passes through v a finite number of times. Then, let ρ' be a suffix starting from v and not passing through v anymore, let $i \in \{\alpha, \beta\}$ be the index such that E_i contains the first edge of ρ', let $\pi_1, \ldots, \pi_l$ be all the loops from v to v in ρ, starting with edge E_i, and π_0 be the shortest prefix of ρ ending with v. Then, if $i = \beta$ (resp., $i = \alpha$) $(\prod_{j=1}^{l} \pi_l)\rho'$ (resp., $\pi \cdot (\prod_{j=1}^{l} \pi_l)\rho'$) is consistent with τ_i on the arena A_i, hence, it is winning. By prefix independence, ρ' and ρ are both winning.

3. Suppose the path ρ passes through v an infinite number of times, and that there exists an index $i \in \{\alpha, \beta\}$ such that ρ' passes infinitely often through edges in E_i but not through edges in $v^{\rightarrow} \setminus E_i$. Then, let ρ' be a suffix starting from v and not using any edge in $v^{\rightarrow} \setminus E_i$. Let π be the prefix of ρ before ρ' such that $\rho = \pi \cdot \rho'$. Let $\pi_1, \ldots, \pi_l$ be all the loops from v to v in π, starting with edge E_i, and π_0 be the shortest prefix of π ending with v. Then, if $i = \beta$ (resp., $i = \alpha$) $(\prod_{j=1}^{l} \pi_l)\rho'$ (resp., $\pi \cdot (\prod_{j=1}^{l} \pi_l)\rho'$) is consistent with τ_i on the arena A_i, hence, it is winning. By prefix independence, ρ' and ρ are both winning.

4. Suppose the path ρ passes through v an infinite number of times, and uses infinitely often edges from both the sets E_α and E_β. Let ρ' be a suffix of ρ starting from v, for $i \in \{\alpha, \beta\}$ let $\pi_{i,1}, \ldots, \pi_{i,l}, \ldots$ be the infinite sequence of loops from v to v on ρ', such that the loops start with an edge in E_i and are written in the order as they appear in ρ'. Then, if $i = \beta$ (resp., $i = \alpha$) the path $\rho_i = (\prod_{j=1}^{+\infty} \pi_{i,j})$ (resp., $\rho'_i = \pi_u \cdot (\prod_{j=1}^{+\infty} \pi_{i,j})$) is consistent with τ_i on A_i and, hence, is winning. Since ρ' lies in the shuffle of ρ_α and ρ_β by convexity, ρ' is winning, and by prefix independence ρ is winning as well.

□

Due to the almost commutativity of the shuffle operation, as corollary of the previous theorem we obtain that the union of winning sets of two strategies is a winning set of the shuffle.

Corollary 4.4.1. *Let (A_α, A_β) be a v-partition of an arena $A = (V_0, V_1, E)$ with respect to E_α, E_β, and W be a prefix-independent and convex goal. Let $\tau_i = (M_i, \psi_i, \tau_{\mu,i}, \mu_{0,i})$ be a strategy for player 0 on A_i and let $D_i \subseteq V_0 \cup V_1$ be a winning set for τ_i on A_i. Let $\tau = \tau_\alpha \otimes_v \tau_\beta = (M, \psi, \tau_\mu, \mu)$ be the shuffle strategy. If v is contained in both the winning sets D_α and D_β, then, $D = D_\alpha \cup D_\beta$ is the winning set set for $\tau_D = (M, \psi, \tau_\mu, \mu_D)$ where μ_D is such that:*

1. *for all $u \in D_\alpha$, w have $e\mu_D(u) = (\alpha, \mu_{0,\alpha}(u), \mu_{0,\beta}(v))$*
2. *for all $u \in D_\beta \setminus D_\alpha$, we have $\mu_D = (\beta, \mu_{0,\alpha}(v), \mu_{0,\beta}(u))$.*

Proof. Consider the shuffle $\tau = \tau_\alpha \otimes_v \tau_\beta = (M, \psi, \tau_\mu, \mu_0)$ and the shuffle $\tau' = \tau_\beta \otimes_v \tau_\alpha = (M, \psi, \tau, \mu'_0)$. By construction the starting memory function are such that for all $u \in V_0 \cup V_1$ we have $\mu_0(u) = (\alpha, \mu_{0,\alpha}(u), \mu_{0,\beta}(v))$ and $\mu'_0(u) = (\beta, \mu_{0,\alpha}(v), \mu_{0,\beta}(u))$.

Since τ_α and τ_β are winning at the node v, by Lemma 4.4.2, it holds that τ is winning on every node in D_α and τ' is winning on every node in D_β. The two strategies differs just for the starting memory. Hence, the strategy $\tau_D = (M, \psi, \tau_\mu, \mu_D)$ behaves like τ when the play starts at a node $u \in D_\alpha$ and behaves like τ' when the play starts at a node $u \in D_\beta \setminus D_\alpha$. So, every play starting at a node $u \in D$ and consistent with τ_D is winning.

□

In the following section such a corollary hold because the fair and priority goals we treat are prefix-independent and convex.

4.4.3 Inductive Step

As inductive case, consider an arena $A = (V_0, V_1, E)$ such that there are n edges exiting from nodes controlled by player 1. Furthermore, suppose we are able to construct winning strategies in memory-update form for all arenas with less than n edges exiting from nodes controlled by player 1. Let $v \in V_1$ be a node of player 1 from which there are at least two exiting edges. Let E_α, E_β be a partition of the set $v^\rightarrow$ in two non-empty and disjoint sets. For $i \in \{\alpha, \beta\}$, let $A_i = (V_0, V_1, E'_i)$ be the subarena of A obtained by removing all edges exiting from v but the edges in E_i. Then, each arena A_i contains less than n edges exiting from nodes controlled by player 1. Then, by induction hypothesis, we know the following informations on A_i.

1. A strategy $\tau_i = (M_i, \psi_i, \tau_{\mu,i}, \mu_{0,i})$ in memory update form and the winning set $D_i \subseteq V_0 \cup V_1$ for τ_i on A_i.
2. For each $u \notin D_i$, there exists a winning strategy for player 1 on A_i at u.

Observe first that for all $u \notin D_\alpha \cap D_\beta$ there exists a winning strategy for player 1 on A at u. Indeed, suppose $u \notin D_i$, then there exists a winning strategy τ_1 for player 1 on the arenas A_i. Since such a strategy always chooses edges in E_i at v , player 1 can force every play to develop in A_i and be consistent with τ_1. Hence, such plays are all losing and τ_1 is a winning strategy for player 1 on A at u. Hence, the maximal winning set is not bigger than $D_\alpha \cap D_\beta$.

Observe also that if $v \notin D_\alpha$ then it must be $D_\alpha \subseteq D_\beta$. Indeed, every play starting at $u \in D_\alpha$ and consistent with τ_α on A_α does not pass through v. Since, A_α and A_β differ only for edges

exiting from v, every path starting at u is consistent with τ_α on A_α if and only if it is consistent with τ_α on A_β. Hence, τ_α is also winning on A_β at u, which means u is in the winning set D_β. By symmetry, if $v \notin D_\beta$ then it must be $D_\beta \subseteq D_\alpha$.

Depending on whether the node v appears in all sets D_i or not, the winning strategy of player 0 on A is constructed in two ways.

1. Suppose there exists a j such that $v \notin D_j$. Without loss of generality, suppose $v \notin D_\alpha$. By Lemma 4.4.1, starting from a node $u \in D_\alpha$, player 0 can force the game to never pass through v by using the strategy τ_α. Let τ be a strategy that behaves like τ_α on all paths in A_α starting from a node in D_α and behaves arbitrary on other paths. On the arena A, starting from a node in D_α, τ will never allow passage through node v. Hence, the game only develops on A_α with strategy τ_α and the constructed path is winning for player 0. Thus, τ is a winning strategy on the set of nodes $D = D_\alpha$. Since the maximal winning set of nodes is $D_\alpha \cap D_\beta$, we have that $D_\alpha \subseteq D_\alpha \cap D_\beta \subseteq D_\alpha$. Hence, $D_\alpha = D_\alpha \cap D_\beta$ is also the maximal winning set.

2. Suppose for all indexes $i \in \{\alpha, \beta\}$, the set D_i contains v. Let $\tau = \tau_\alpha \otimes_v \tau_\beta$ be the shuffle strategy and $(M, \psi, \tau_\mu, \mu_0)$ be its memory-update form. By Corollary 4.4.1 the strategy $\tau_D = (M, \psi, \tau, \mu_D)$ is a winning strategy for player 0 on A at every node in $D_\alpha \cup D_\beta$. Since the maximal winning set of nodes is $D_\alpha \cap D_\beta$, we have that $D_\alpha \cup D_\beta \subseteq D_\alpha \cap D_\beta \subseteq D_\alpha \cup D_\beta$. Hence, $D_\alpha \cup D_\beta = D_\alpha \cap D_\beta$ is also the maximal winning set.

This concludes both the construction and the correctness proof of the algorithm for the computation of the winning strategy for player 0.

4.4.4 Complexity

Let $A = (V_0, V_1, E)$ be an arena in input to our algorithm. Let n be the number of nodes in V_1 from which there are least two outgoing edges, and $m = \max\{|v^\rightarrow| \mid v \in V_1\}$ be the maximum number of edges exiting from nodes of player 1. The base case is polynomial in the size of the arena and requires time and space $P(A)$. The inductive step is called at most $\log_2(|v^\rightarrow|)$ for each node $v \in V_1$, hence the whole algorithm is executed in time exponential in $\sum_{v \in V_1} \log_2(|v^\rightarrow|)$. Moreover, in the worst case the strategy's size doubles at every induction call, hence, the solution and the whole algorithm require space exponential in $\sum_{v \in V_1} \log_2(|v^\rightarrow|)$. The maximum depth of an induction call is $\sum_{v \in V_1} \log_2(|v^\rightarrow|) \leq n \cdot log_2(m)$, hence every base call requires at most time and space polynomial $P(A) + P(n) = P'(A)$ in the arena, hence the algorithm require space and time $P'(A)^{\sum_{v \in V_1} \log_2(|v^\rightarrow|)}$. Since, $n \leq \sum_{v \in V_1} \log_2(|v^\rightarrow|) \leq n \cdot log_2(m)$, the algorithm requires time and space exponential in the number of nodes controlled by player 1 from which there are at least two outgoing edges.

By slightly modifying the algorithm we can reduce the memory space the result strategy. Recall that the shuffle strategy τ at a node v of two strategies τ_α and τ_β has memory space $M = \{\alpha, \beta\} \times M_\alpha \times M_\beta$. However, by construction of the memory-update function ψ, the first component of the memory passes from α to β or vice-versa only at the node v. This means that when the first component is α (resp., β), then the second component of memory $m_\beta \in M_\beta$ (resp.,

second component of the memory $mn_\alpha \in M_\alpha$) is never updated and is fixed to the memory of τ_β (resp., τ_α) of a path ending with v. The memory is sensible to the ending node, because a choice function τ_μ associates to each memory an edge starting from that ending node, hence two paths with different ending node cannot have same memory. Hence, instead of using a memory space $M = \{\alpha, \beta\} \times M_\alpha \times M_\beta$ we can use a smaller space $M = \{\alpha\} \times M_\alpha \times M_{\beta,v} \cup \{\beta\} \times M_{\alpha,v} \times M_\beta$, where $M_{i,v} \subseteq M_i$ is the set of memories assumed by τ_i on all and only the paths ending in v. Observe, that in order to use such a smaller space, we need to propagate inductively the informations on the set of memories assumed by paths ending with any given node of player 1. This is done trivially in the base case, and for the inductive case, the memory of paths ending in node of u is $\{\alpha\} \times M_{\alpha,u} \times M_{\beta,v} \cup \{\beta\} \times M_{\alpha,v} \times M_{\beta,u}$.

However, the above improvement on the memory space still admits that in the worst case the resulting strategy has exponential size. It is sufficient that at every inductive step $M_{i,v}$ is close to some fraction of M_i. For example if $M_{i,v} = \frac{1}{|V|} M_i$, and in the base case the memory space is M, the resulting memory space at worst has size $(\frac{2}{|V|})^{\sum_{v \in V_1} \log_2(|v^\rightarrow|)} (|M|)^{2^{\sum_{v \in V_1} \log_2(|v^\rightarrow|)}}$.

...

...

5 Half-Positionality

Contents

5.1 Introduction

Half-positionality is an essential property for the proof of the results in the previous chapter. By determining that, in a system-environment game, the environment winning capability is not weakened by using only positional strategies, we are able to evaluate whether the system can win on all the subgraphs induced by such positional strategies. Hence, half-positionality provides a $Co - NP$ algorithm that determines whether there exists a winning strategy for the system. Hence, in general half-positionality is useful to determine the winning player in a game. For example, it may be used in a prover-disprove game to determine the validity or the contradiction of a logic formula. It can also be used in other system-environment games with respect to goals such as other fairness and priority conditions different from those discussed in this thesis. Moreover, when in a system environment game, one proves that the game is half-positional for the system, the result is also more interesting. Indeed, in this case, the system can satisfy the specification by using a very simple strategy that allows an easy computation of the decision to take at every state. However, Kopczyński's theorem is not a characterization and, hence, there are goals whose half-positionality is difficult to determine. In this chapter, we investigate half-positionality with the aim to find a characterization, however, we determine only a novel sufficient condition which is broader than the one defined by Kopczyński as long as determined goals are considered. By Borel Determinacy Theorem ([Mar75], [Bry01]), all Borel games are determined. Since only a few exotic goals, hard to meet in practical applications, are not Borel, we believe our result is an effective mean to prove half-positionality when Kopczyński's hypothesis cannot be applied.

5.2 Preliminaries

In the following we use order relations on words: for a goal W and a pair of words $x, y \in [k]^\omega$, we say that *(i)* x is not better than y when they are both losing or y is winning, and we write $x \leq_W y$, *(ii)* y is better that x, when y is winning and x is losing, and we write $x <_W y$. In the same way, for a goal W and a pair of sets $M, N \in [k]^\omega$ *(i)* we say that M is not better than N, i.e. $M \leq_W N$, to mean that if M contains a winning word then N contains a winning word too, and *(ii)* we say that N is better than M, i.e. $M <_W N$, to mean that M contains only losing words and N contains at least a winning word. For ease of reading, when the goal W is clear from the contest, we write $x < y$, $x \leq y$, $M < N$ and $M \leq N$, respectively, for $x <_W y$, $x \leq_W y$, $M <_W N$ and $M \leq_W N$. With the following two lemmas, we reformulate the definition of concavity and prefix-independence in terms of languages, rather than of single words.

Lemma 5.2.1. *A goal $W \subseteq [k]^\omega$ is prefix-independent if and only it for all color sequences $x \in [k]^*$ and sets of color sequences $M \subseteq [k]^\omega$ we have that $xM \leq M$ and $M \leq xM$.*

Proof. Suppose that W is prefix-independent. If M contains a winning word m, then xM contains the winning word xm, and we have both $xM \leq M$ and $M \leq xM$. If M contains only losing words m, then xM contains only losing words xm and we have both $xM \leq M$ and $M \leq xM$.

Suppose now that, for all languages $M \subseteq [k]^\omega$, we have $xM \leq M$ and $M \leq xM$. Moreover, suppose by contradiction that $W \neq xW$. Then, there exists a word m such that $xm \notin W$. Hence, for the language $M = \{m\}$ we do not have $M \leq xM$.

□

Lemma 5.2.2. *A goal $W \subseteq [k]^\omega$ is concave if and only if for all languages $M, N \subseteq [k]^\omega$ we have that $M \otimes N \leq M \cup N$.*

Proof. Suppose that W is concave. For all $M, N \subseteq \overline{W}$, we have that $M \otimes N \subseteq \overline{W}$. So, for all languages $M, N \in [k]^\omega$, if M or N contains a winning word in W, we have in both cases $M \otimes N \leq M \cup N$; conversely, if M and N contain only losing words, by hypothesis, so does $M \otimes N$. Hence, we have that $M \otimes N \leq M \cup N$.

Suppose now that for all languages $M, N \subseteq [k]^\omega$ we have $M \otimes N \leq M \cup N$. Then, if M and N contain only losing words, $M \otimes N$ must contain only losing words too. Thus, for all $M, N \in \overline{W}$ we have that $M \otimes N \subseteq \overline{W}$.

□

5.2.1 Gimbert and Zielonka's Theorem

Before starting our investigation about a possible characterization of half-positionality, it is useful to recall a similar result due to Gimbert and Zielonka [GZ05] about full-positionality. Such a result does not only allow us to prove the positionality of some goals, but give us inspiration for the novel sufficient condition and for a possible characterization.

When player i uses a memoryless strategy, we can construct a subgraph that is obtained by removing all the edges not used by that strategy, from the nodes controlled by player i. In such subgraph, there is only one exiting edge from every node of player i. Hence, player $1 - i$ actually makes choice and constructs a path on a graph. Since, the graph may considered as the representation of a finite state automaton, the color sequence of a path constructed by player $i - 1$ belongs to a regular language. Also the finite paths, from a node of player $i - 1$ to itself belong to a regular language. Hence, Gimbert and Zielonka observed that in order to obtain the half-positionality for player $i-1$, they do not need to ask that player $i-1$ does not prefer switching between different arbitrary complex behaviors, but just that player $i - 1$ does not prefer switching between behaviors composed by some regular languages. So, by focusing the attention on a node v where player $i-1$'s choices should not alternate, we observe that an infinite path may be composed by a finite or infinite number of loops through that node and an infinite path starting from that node and never coming back. We can identify the languages of color sequences of the loops, and the language of color sequence of infinite path starting from v. We obtain different loop-language depending on the first choice of player $i - 1$ at the node v. In order to evaluate the infinite words by means of sets of regular languages that contain only finite words, Gimber and Zielonka extends the finite set of words to infinity through a limit on the words.

Definition 5.2.1. Infinite Extension

Let $M \subseteq [k]^*$ be a language of finite words, then the infinite extension of M is the set $\langle M \rangle$ of infinite words $x \in [k]^\omega$ whose prefixes are prefixes of at least one word in M.

Then, Gimbert and Zielonka state that player $i - 1$ does not prefer switching between two behaviors through the property of selectivity. It states that, given two languages M and N of loops from v to v and a language K of paths from v, switching infinitely often between the two sets of

loops or switching finitely often and then progressing with a path in K is not better than using always the same set of loops infinitely often or starting directly with a path in K. Observe that our interpretation holds because $\langle (M \cup N)^* K \rangle = \langle (M \cup N)^* \rangle \cup (M \cup N)^* \langle K \rangle$

Definition 5.2.2. *Selectivity*

A goal W is selective *if and only if for all $x \in [k]^*$ and for all regular languages $M, N, K \subseteq [k]^*$ we have that $x\langle (M \cup N)^* K \rangle \leq x\langle M^* \rangle \cup x\langle N^* \rangle \cup x\langle K \rangle$.*

However, selectivity does not avoid that player $i-1$'s choice depends on some finite prefix up to a node. Hence, like in Kopczyńsky result, one may think to use the prefix-independent property. However, Gilbert and Zielonka observe that such a strong property is not needed. Even if on some node the choice depend on a finite prefix, we can still define an order relation on the choices and determine which one behaves better for all the possible prefixes.

Definition 5.2.3. Definitive Order Relation

Given two languages of finite words $M, N \subseteq [k]^\omega$, N is definitively better than M if for all prefixes $x \in [k]^*$, it holds that $xM \leq xN$. We write it $M \sqsubseteq N$.

When player $i-1$ at a given points needs to choose between two or more behaviors, it is sufficient to take the one which is definitively better than others. Indeed, no matter the prefix up to that point, that choice gives better results. However, the relations needs to be total otherwise a dominant behavior may not exists. Since a language is regular if and only if there exists a n automaton graph that recognizes it, on graph controlled by player $i-1$ the relation just needs to be total among infinite extensions of sets of regular languages.

Definition 5.2.4. Monotonicity

A goal $W \subseteq [k]^\omega$ is monotone *if and only if for all pairs of regular languages $M, N \subseteq [k]^*$ it holds $\langle M \rangle \sqsubseteq \langle N \rangle$ or $\langle N \rangle \sqsubseteq \langle M \rangle$.*

Actually Gimbert and Zielonka state monotonicity by means of an another definition, whose equivalence to the above one is proved by the following lemma.

Lemma 5.2.3. *A goal $W \subseteq [k]^*$ is* monotone *if and only if for all words $x \in [k]^*$ and all regular languages $M, N \subseteq [k]^\omega$ it holds that $x\langle M \rangle < x\langle N \rangle$ implies that for all $y \in [k]^*$ it is $y\langle M \rangle \leq y\langle N \rangle$.*

if. Suppose $\langle M \rangle \not\sqsubseteq \langle N \rangle$, then there exists $x \in [k]^*$ such that $x\langle N \rangle < x\langle M \rangle$, hence, by hypothesis, for all $y \in [k]^*$ we have $y\langle N \rangle \leq y\langle M \rangle$. So $\langle N \rangle \sqsubseteq \langle M \rangle$ and the thesis holds.

[only if] Suppose by contradiction that $\langle M \rangle \not\sqsubseteq \langle N \rangle$ and $\langle N \rangle \not\sqsubseteq \langle M \rangle$, then there exist $x, y \in [k^*]$ such that $x\langle N \rangle < x\langle M \rangle$ and $y\langle N \rangle < y\langle M \rangle$ and this contradicts the hypothesis.

□

The two previous property constitute a characterization to full-positionality. The following is just a corollary of the more general result from Gimbert and Zielonka, who actually state it in the more general framework of optimization games.

Theorem 5.2.1. Full-positionality characterization [GZ05]

A winning condition $W \subseteq [k]^\omega$ is full-positional if and only if both W and $[k]^\omega \setminus W$ are selective and monotone.

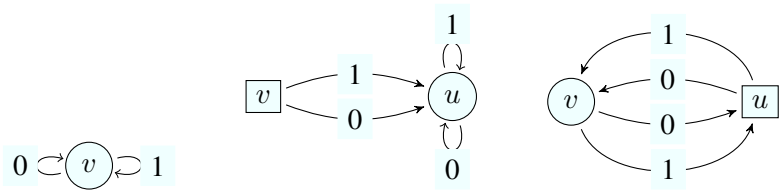


Figure 5.1: Three game arenas.

5.3 Strong monotonicity and strong concavity

Kopzyńsky's theorem is not a characterization for half-positionality, but Gimbert and Zielonka could define a characterization for full-positionality by relaxing concavity and prefix independence to selectivity and monotonicity. Although we can no longer operate with regular languages, since one player is allowed to use a non-positional strategy, it is still reasonable to ask whether we can relax concavity and prefix independence to some other properties still sufficient for half-positionality. Our investigation starts from the counter-example showing that a half-positional goal does not need to be prefix-independent.

Example 5.3.1. *Consider the winning condition $W = 0(1^*0)^\omega + 1[1]^\omega$. It contains all and only the words x such that either x starts with color 1, or x starts with color 0 and contains infinitely many 0. The goal is not prefix-independent: if $y \in [1]^\omega$ does not contain infinite 0 then $0 \cdot y$ is losing and $1 \cdot y$ is winning. However, the goal is half-positional, more exactly it is full-positional, and we prove this through Gimbert and Zielonka's theorem. Precisely we show that W and $W' = [1]^\omega - W = 0[1]^*1^\omega$ are both monotone and selective.*

1. *W is selective. Consider $M, N, K \subseteq [1]^*$ and $x \in [k]^*$. Suppose that the set $x\langle M^*\rangle \cup x\langle N^*\rangle \cup x\langle K\rangle$ contains only losing words. Then there are two possible situations:* (i) *x is not empty and starts with color 0, or* (ii)*x is empty and all words in M, N, K starts with color 0. In both cases, necessarily no word in M or N contains occurrences of color 0 and $\langle K\rangle$ does not contain words with infinitely many 0. Hence, $x\langle(M \cup N)^*K\rangle$ contains only words starting with color 0 and containing finitely many occurrences of color 0. Thus $x\langle(M \cup N)^*K\rangle$ is never better than $x\langle M^*\rangle \cup x\langle N^*\rangle \cup x\langle K\rangle$.*

2. *W' is selective. Consider $M, N, K \subseteq [1]^*$ and $x \in [k]^*$. Suppose that the set $x\langle M^*\rangle \cup x\langle N^*\rangle \cup x\langle K\rangle$ contains only losing words.*(i) *Suppose x starts with color 0, or x is empty and there exists at least a word in M, N, K starting with color 0. Then, all words in M and N contains at least one occurrence of color 0, and all words in $\langle k\rangle$ contains infinitely many occurrences of color 0. So every word in the set $x\langle(M \cup N)^*K\rangle$ starts with 0 and contain infinitely many zero and is losing with respect to W'.* (ii) *Suppose that x starts with color 1, or all words in M, N, K start with color 1. Then all words in $x\langle(M \cup N)^*K\rangle$ are losing.*

3. *W is monotone. Consider $M, N \subseteq [1]^*$ and $x \in [1]^*$ such that $x\langle M\rangle <_W x\langle N\rangle$. Then x starts with color 0 else both sets are winning. Moreover, $\langle M\rangle$ contains only words with finitely many occurrences of color 0 and $\langle N\rangle$ contains a word with infinite occurrences of*

color 0*. Consider a new word* $y \in [1]^*$*, if* y *starts with color* 0 *we still have* $y\langle M\rangle <_W y\langle N\rangle$*. If* y *starts with color* 1*, both sets* $x\langle M\rangle, x\langle N\rangle$ *are winning. So in both cases* $x\langle M\rangle \leq_W x\langle N\rangle$*.*

4. W'' *is monotone. Consider* $M, N \subseteq [1]^*$ *and* $x \in [1]^*$ *such that* $x\langle M\rangle <_W x\langle N\rangle$*. Then* x *starts with color* 0 *else both sets are losing. Moreover,* $\langle M\rangle$ *contains only words with infinitely many occurrences of color* 0 *and* $\langle N\rangle$ *contains a word with finite occurrences of color* 0*. Consider a new word* $y \in [1]^*$*, if* y *starts with color* 0 *we still have* $y\langle M\rangle <_W y\langle N\rangle$*. If* y *starts with color* 1*, both sets* $x\langle M\rangle, x\langle N\rangle$ *are losing. So in both cases* $x\langle M\rangle \leq_W x\langle N\rangle$*.*

This counter-example shows that, like discussed in the introduction to monotonicity, prefix independence is a strong property which is not needed for positionality. Indeed, even if the winning nature of the decision of a player on a node depends on the prefix up to that node, it is still possible that a decision performs better than the others on all prefixes. Then, like in the monotonicity property, we need to ask that given two decision for a given player, there do not exist two prefixes that completely change the winning nature of the two decisions. Since the behavior of a player is no longer ensured to be regular by the use of a positional strategy by the other player, we make use of a strong version of monotonicity that takes into account also non-regular behaviors.

Definition 5.3.1. Strong Monotonicity

A goal $W \subseteq [k]^\omega$ *is* strongly monotone *if and only if for all pairs of infinite languages* $M, N \subseteq [k]^\omega$ *it holds* $M \sqsubseteq N$ *or* $N \sqsubseteq M$*.*

Like monotonicity, strong monotonicity can be defined equivalently as follows.

Lemma 5.3.1. *A goal* $W \subseteq [k]^\omega$ *is strongly monotone if and only if, for all words* $x \in [k]^*$ *and languages* $M, N \subseteq [k]^\omega$*, it holds that* $xM < xN$ *implies that for all* $y \in [k]^*$ *it is* $yM \leq yN$*.*

As one may easily guess, strong monotonicity is a weaker property compared to prefix-independence, as shown by the following lemma.

Lemma 5.3.2. *All prefix-independent goals are strongly monotone. Moreover, there is a goal which is strongly monotone, but not prefix-independent.*

Proof. For the first part, we have by hypothesis that, for all $x \in [k]^*$, and $M \subseteq [k]^\omega$, it holds that $M \leq xM \leq M$. Now, take two languages $M, N \subseteq [k]^\omega$, and suppose that there exists an $x \in [k]^*$ such that $xM < xN$, then for all $y \in [k]^*$ we have $yM \leq M \leq xM \leq xN \leq N \leq yN$.

For the second part, let $k = 1$, a strongly monotone and prefix-dependent goal is given by the language of all words containing at least one 0, i.e., $W = [k]^*0[k]^\omega$. It is easy to see that the goal is not prefix-independent, because the word 1^ω is losing while the word 01^ω is winning. We show that the goal is strongly monotone. Consider two languages $M, N \subseteq [k]^*$, and suppose that there exists an $x \in [k]^*$ such that $xM < xN$, then xN contains a winning word and xM contains only losing words. Observe first that x cannot contain 0, or else all words in xM would be winning. So $x \in 1^*$, there exists a word in N that contains 0, and all words in M contain only 1's. So, for each $y \in [k]^*$, there is always a word in yN containing 0. Since yN contains a winning word, we have $yM \leq yN$.

□

Strong monotonicity seems a good substitute to prefix-independence. Unfortunately, the following lemma shows that strong monotonicity cannot replace prefix-independence in the hypotheses of Theorem 4.2.1.

Lemma 5.3.3. *There is a strongly monotone and concave goal which is not half-positional.*

Proof. For $k = 1$, the strongly monotone and concave goal is $W = [k]^*01^\omega$. We prove first that the goal is strongly monotone and concave. A word is losing if and only if it is either 1^ω or it does not have 1^ω as a suffix. Let $x \in [k]^*$, $n, m \in [k]^\omega$ with $xn, xm \notin W$. There are two situations to discuss. First, assume that x does not contain 0. Then, n and m may be both 1^ω in which case $x(m \otimes n) = 1^\omega$ or at least one between n and m contains 0 infinitely often, thus the shuffle of n and m contains only words that pick colors from both the sequences infinitely often and thus only words that contain 0 infinitely often. So, $x(m \otimes n)$ contains losing word even in this case. Instead, assume that x contains 0. Then, n and m contain 0 infinitely often and the same reasoning above applies. So the goal is concave. Let $x \in [k]^*$, $n, m \in [k]^\omega$ such that $xm \notin W$ and $xn \in W$. We prove strong monotonicity by showing that for all $y \in [k]^*$ it holds that $ym \notin W$ or $yn \in W$. We again distinguish two cases. First, assume that x does not contain a 0. Then, n contains 0 and a suffix 1^ω thus for every $y \in [k]^*$, we have $yn \in W$ since it contains 0 and a suffix 1^ω. Instead, assume that x contains a 0. Then, m contains 0 infinitely often, thus for every $y \in [k]^*$ the word $ym \notin W$ since it contains 0 infinitely often. The above goal is not half-positional in the following arena $(\{v\}, \emptyset, v, \{(v, 0, v), (v, 1, v)\})$ (Fig. 5.1(a)), in such a game graph player 0 wins by choosing at least once the edge with color 0 and then always the edge with color 1.

□

Observe that, in the previous counterexample, the key element that does not allow half positionality is the fact that player 0 prefers switching between two different behaviors finitely often and then progressing indefinitely along one of them. However, concavity just requires that player 0 prefers following a fixed behavior rather than switching between two different ones *infinitely often*. Thus, we introduce a modification to the property of concavity, requiring not only that alternating infinitely often between two losing words yields a losing word, but also that alternating *finitely* often between two losing words and then progressing along one of them yields a losing word.

Definition 5.3.2. Strong Shuffle

For two color sequences $x, y \in [k]^\omega$, the strong shuffle *of x and y, denoted by $x \otimes_s y$ is the language containing*

1. *the set $x \otimes y$;*
2. *the words $z_1 z_2 \dots z_l z' \in [k]^\omega$, for odd l, $z_i \in [k]^*$ and $z' \in [k]^\omega$, such that it holds $x = z_1 z_3 \dots z_l z'$ and $y = z_2 z_4 \dots z_{l-1} y'$, for some $y' \in [k]^\omega$;*
3. *the words $z_1 z_2 \dots z_l z' \in [k]^\omega$, for even l, $z_i \in [k]^*$ and $z' \in [k]^\omega$, such that it holds $x = z_1 z_3 \dots z_{l-1} x'$ and $y = z_2 z_4 \dots z_l z'$, for some $x' \in [k]^\omega$.*

For two languages $M, N \subseteq [k]^\omega$, the strong shuffle *of M and N is the set $M \otimes_s N = \cup_{n \in N, m \in M} (m \otimes_s n)$.*

Definition 5.3.3. Strong Concavity

A goal $W \subseteq [k]^\omega$ is strongly concave *if and only if, for all words $x \in [k]^*$, $n, m \in [k]^\omega$, and $z \in x(m \otimes_s n)$, it holds that if $z \in W$ then either $xn \in W$ or $xm \in W$.*

It is immediate to see that a strongly concave goal is concave too. In the following, we make use of an equivalent definition of strong concavity that operates on languages.

Lemma 5.3.4. *A goal $W \subseteq [k]^\omega$ is strongly concave if and only if, for all words $x \in [k]^*$ and languages $M, N \subseteq [k]^\omega$, it holds that $x(M \otimes_s N) \leq xM \cup xN$.*

Hence, by making use of the relation $\sqsubseteq$, we obtain that strong concavity expresses the fact that two words are definiteively better than their strong shuffle.

Lemma 5.3.5. *A goal $W \subseteq [k]^\omega$ is strongly concave if and only if, for all languages $M, N \subseteq [k]^\omega$, it holds that $(M \otimes_s N) \sqsubseteq M \cup N$.*

Even the property of strong concavity is not sufficient to ensure half positionality.

Lemma 5.3.6. *There is a strongly concave goal which is not half-positional.*

Proof. For $k = 1$ the strongly concave goal is $W = 0^\omega \cup 1^\omega$. Two losing words n and m contain at least an occurrence of the color 1 and an occurrence of the color 0, thus every word in their strong shuffle will contain at least an occurrence of color 1 and an occurrence of color 0 and it will be losing. So the strong concavity of the goal is proved. The above goal is not half-positional in the following 2-colored arena $(\{u\}, \{v\}, v, \{(v, 0, u), (v, 1, u), (u, 0, u), (u, 1, u)\})$, showed in Figure 5.1(b). In this arena player 0 wins the game by choosing forever the edge $(u, 0, u)$ or the edge $(u, 1, u)$ depending on what color was chosen by player 1 to reach u from v.

□

In the previous counterexample, by choosing a different prefix, player 1 can exchange the winning nature of the following choices of player 0. That is why strong monotonicity is essential since it somehow allows player 0 to operate while forgetting the past decisions taken by player 1.

We argue now that the two introduced properties of strong monotonicity and strong concavity are strictly less restrictive than the properties of prefix independence and concavity.

Lemma 5.3.7. *Concave and prefix-independent winning conditions are strongly monotone and strongly concave.*

Proof. By Lemma 5.3.2 we already have that a prefix-independent goal is strongly monotone. It remains to show that a concave and prefix-independent goal is strongly concave.

For a language $M \subseteq [k]^\omega$, let $\mathit{suff}(M)$ and $\mathit{pref}(M)$ be the sets of suffixes and prefixes of words in M, respectively. By concavity, for all $M, N \subseteq [k]^\omega$ we have $M \otimes N \leq M \cup N$ and by prefix independence we have for all $M \in [k]^\omega$ and for all $x \in [k]^*$ $M \leq xM \leq M$. Take any word $x \in [k]^*$, and any two languages $M, N \subseteq [k]^\omega$. Then we have $x(M \otimes_s N) = x(M \otimes N) \cup x \cdot \mathit{pref}(M \otimes N) \cdot \mathit{suff}(N) \cup x \cdot \mathit{pref}(M \otimes N) \cdot \mathit{suff}(M)$. First, by prefix independence and then by concavity we have $x(M \otimes N) \leq M \otimes N \leq M \cup N \leq x(M \cup N) = xM \cup xN$. Then, $x \cdot \mathit{pref}(M \otimes N) \cdot \mathit{suff}(T) \leq \mathit{suff}(T) \leq xT \leq xM \cup xN$, where $T \in \{M, N\}$. So, we have $x(M \otimes_s N) \leq xM \cup xN$.

□

Lemma 5.3.8. *There exists a strongly monotone and strongly concave goal which is not prefix independent.*

Proof. Let $k = 1$, the goal is given by the set of words that either start with 1, or start with 0 and contains infinitely many 0's, i.e., $W = 0(1^*0)^\omega \cup 1[k]^*$. It is easy to see that the goal is not prefix-independent: indeed, for $M = 1^\omega$ we have that $0M \leq M$, but not $M \leq 0M$ since M contains only winning words and $0M$ only losing ones.

Next, we prove that the goal is strongly monotone. Consider $M, N \subseteq [k]^*$ and $x \in [k]^*$ and suppose that $xM < xN$, so xN contains a winning word and xM contains only losing ones. Observe that x does not start with 1, otherwise all words in xM would be winning. So, there are two situations to discuss: $x = \varepsilon$ or x starts with 0. If $x = \varepsilon$ then all words in M starts with 0 and have a suffix equal to 1^ω. Now for all $y \in 1[k]^*$ we have $yM \leq yN$ since all the words in all languages are winning; for all $y \in 0^*[k]^*$ we have $yM \leq yN$ because all the words in yM are losing since they start with 0 and have a suffix 1^ω. If instead x starts with 0 then there exists a word $n \in N$ that contains infinitely many 0, for every $y \in [k]^*$ the word yn will contain infinitely many 0 and it will be winning, thus for all $y \in [k]^*$ we will have $yM \leq yN$.

Now we prove that the goal is strongly concave. Consider $x \in [k]^*$, $M, N \subseteq [k]^\omega$ and $K \subseteq [k]^*$. We want to prove that $x(M \otimes_s N) \leq xM \cup xN$. If the r.h.s. of the inequality contains a winning word, the inequality trivially holds. So, suppose that the r.h.s. does not contain a winning word, so it cannot be $x \in 1[k]^*$ but it must be $x \in 0[k]^* \cup \{\varepsilon\}$. If x starts with 0, every word in M, N contains a suffix 1^ω and all words in $M \otimes_s N$ contain a suffix 1^ω. So, $M \otimes_s N$ contains only losing words. If $x = \varepsilon$, every word in M, N contains a suffix 1^ω and starts with 0, so all words in $M \otimes_s N$ contain a suffix 1^ω and start with 0, and therefore they are losing. □

5.4 Half-positionality theorem

In this section, we prove that determinacy, strong monotonicity and strong concavity are sufficient but not necessary conditions to half positionality for player 0.

Theorem 5.4.1. *A determined, strongly monotone and strongly concave goal is half-positional.*

Proof. The proof proceeds by induction on the number of edges exiting from the nodes controlled by player 0 in the game arena. As a base case in the graph G for each node controlled by player 0 there exists only one exiting edge. In such a graph player 0 has only one possible strategy which is positional. So, the result is trivially true. Suppose that in the arena there are n edges exiting from nodes of player 0 and that, for all graphs with at most $n - 1$ edges exiting from nodes of player 0, if player 0 has a winning strategy he has a positional one. Let t be a node of player 0 in G such that there is more than one edge exiting from t. We can partition the set of edges exiting from t in two disjoint non-empty sets E_α and E_β. Let G_α and G_β be the two subgraphs obtained from G by removing the edges of E_β and E_α, respectively. There are two cases to discuss.

First, suppose that in G_α or G_β player 0 has a winning strategy. Then, by inductive hypothesis he has a positional winning strategy. It is easy to see that such a strategy is winning in G too. Indeed, since player 0 controls the node t, he is able to force the play to stay always in G_α or G_β. Suppose now that player 0 has no winning strategy in G_α and in G_β. We prove the thesis by

showing that player 0 has no winning strategy in G. By determinacy, there exist two strategies τ_α and τ_β winning for player 1 in G_α and G_β, respectively.

Let σ be a strategy of player 0 in G, we show that there exists a strategy of player 1 in G winning in G against σ. If one of the plays $P(\sigma, \tau_\alpha)$ or $P(\sigma, \tau_\beta)$ does not pass through t then that play is in G_α and G_β and so it is winning for player 1 who is using his winning strategy on one of the graphs.

Suppose now that both of the above plays pass through t. Let x_α and x_β be respectively the color sequences of the prefixes of $P(\sigma, \tau_\alpha)$ and $P(\sigma, \tau_\beta)$, up to the first occurrence of t. Let M_α and M_β be the sets of color sequences of suffixes after respectively a prefix x_α and x_β of plays consistent respectively with τ_α and τ_β. Observe that $x_\alpha M_\alpha$ and $x_\beta M_\beta$ contain plays consistent respectively with τ_α in G_α and τ_β in G_β, and such plays are losing for player 0. We prove now that either $x_\alpha M_\beta$ or $x_\beta M_\alpha$ contains only losing words for player 0. Indeed, if $x_\alpha M_\beta$ contains a winning word, we have that $x_\alpha M_\alpha < x_\alpha M_\beta$, since $x_\alpha M_\alpha$ contains only plays losing for player 0. Then, by strong monotonicity we have that, for all $y \in C^*$, it holds $yM_\alpha \leq yM_\beta$ and hence $x_\beta M_\alpha \leq x_\beta M_\beta$. Since $x_\beta M_\beta$ contains only losing words, so does $x_\beta M_\alpha$.

Suppose without loss of generality that $x_\beta M_\alpha$ contains only losing words. Then, we construct the strategy τ'_α, which behaves like τ_α on all partial plays which do not have a prefix x_β. When the partial play has a prefix x_β, it behaves like τ_α when it sees x_α in place of x_β. More formally $\tau'_\alpha(x_\beta \pi) = \tau_\alpha(x_\alpha \pi)$, and in the other cases $\tau'_\alpha(\pi) = \tau_\alpha(\pi)$. Let $\tau'_\beta = \tau_\beta$. We construct a strategy τ in G: at the beginning the strategy behaves like τ_β; when the play passes through t, depending on what subgraph the last edge from t chosen by player 0 belongs to, the strategy τ behaves like τ'_α or τ'_β when they are applied only to the initial prefix up to t and all the loops from t to t, where the first edge belongs to G_α or G_β, respectively.

Formally, for all prefixes π that do not pass through t, we have $\tau(\pi) = \tau_\beta(\pi)$; if π_{i,γ_i} is a loop from t to t with first edge in G_{γ_i}, for all prefixes $\pi = x\pi_{1,\gamma_1}, \ldots, \pi_{n,\gamma_n}\pi_\gamma$, we have $\tau'(\pi) = \tau'_\gamma(x(\prod_{\gamma_i=\gamma} \pi_{i,\gamma_i}\pi_\gamma))$. The play $P(\sigma, \tau)$ coincides with $P(\sigma, \tau_\beta)$ up to t, so it has a prefix with color sequence x_β. After that prefix, the play develops in parallel and alternates pieces of two plays: one in G_β consistent with τ_β, and the other in G_α consistent with τ'_α. So, the color sequence of the two suffixes are respectively in M_β and in M_α.[1] Hence, the color sequence of the suffix after x_β of the play $P(\sigma, \tau)$ lies in the shuffle of M_α and M_β. By strong concavity we have that $Col(P(\sigma, \tau)) \in x_\beta(M_\alpha \otimes_s M_\beta) \leq x_\beta M_\alpha \cup x_\beta M_\beta$. Since both $x_\beta M_\alpha$ and $x_\beta M_\beta$ contain only losing words, we have that $Col(P(\sigma, \tau))$ is a losing word for player 0. Hence, for all strategies σ of player 0 there exists a strategy τ of player 1 winning over 0. We conclude that player 0 has no winning strategy.

□

Since strongly concavity implies concavity, the following result states that the conditions appearing as the hypothesis of the previous theorem and of Theorem 4.2.1 are not a complete characterizations for half positional goals.

Lemma 5.4.1. *There exists a goal that is half positional but not concave.*

Proof. Let $k = 1$, the half positional goal is $W = [k]^*1[k]^*1[k]^\omega$. The goal states that player 0 tries to make color 1 occur at least twice. We show that the goal is not concave: let $x = \varepsilon$,

[1] Note that it is possible that one of the two suffixes does not progress indefinitely.

$n, m = 10^\omega$, then we have $xn, xm \notin W$, but $t = 110^\omega \in m \otimes n$ with $xt \in W$, hence the goal is not concave. The goal is half positional because in every point in a play player 0 does not need to look at the past, but just tries to form a path that passes through as many edges colored with 1 as possible. For a more formal proof, we prove that the goal is full-positional, through Gimbert and Zielonka's theorem. Precisely, we prove that W and $\overline{W} = [1]^* \setminus W$ are both selective and monotone. Observe that $\overline{W}$ is the set of all the words contain at most one 1.

1. W is selective. Suppose by contradiction that W is not selective. Then, there exist $x \in [k]^*$ and $M, N, K \subseteq [k]^*$ such that $x\langle(M \cup N)^* K\rangle = x\langle(M \cup N)^*\rangle \cup x(M \cup N)^*\langle K\rangle$ contains a winning word and $x\langle M^*\rangle \cup x\langle N^*\rangle \cup x\langle K\rangle$ contains only losing words. Observe that no word in M or N contains 1, or else if $m \in M \cup N$ contains a 1, $xm^\omega \in x\langle M^*\rangle \cup x\langle N^*\rangle$ contains infinitely many 1's and it is a winning word. So, the words in the set $x\langle(M \cup N)^*\rangle$ do not contain 1 and they are losing. Moreover, since $x\langle K\rangle$ does not contain more than one 1, the words in $x(M \cup N)^*\langle K\rangle$ do not contain more than one 1 and they are all losing too. So, the set $x\langle(M \cup N)^* K\rangle$ contains only losing words, hence a contradiction.

2. W is monotone. Suppose by contradiction that W is not monotone. Then there exist $x, y \in [k]^*$ and $M, N \subseteq [k]^*$ such that $xM < xN$ and $yN < yM$. So, xM and yN contain only losing words, xN and yM contain a winning word. If x contains more than one 1, all words in the first two sets are losing, hence a contradiction. If x contains one 1, then no word in M contains 1. However, there is a winning word in yM, so y contains two 1's. Hence, yN contain only winning words, which is a contradiction. If x does not contain a 1, there is a word in N with two 1's. Hence, yN contains at least a winning word, which is again a contradiction.

3. $\overline{W}$ is selective. Suppose by contradiction that $\overline{W}$ is not selective. Then, there exist $x \in [k]^*$ and $M, N, K \subseteq [k]^*$ such that $x\langle(M \cup N)^* K\rangle = x\langle(M \cup N)^*\rangle \cup x(M \cup N)^*\langle K\rangle$ contains a winning word and $x\langle M^*\rangle \cup x\langle N^*\rangle \cup x\langle K\rangle$ contains only losing words. Observe that no word in M or N does not contain 1, else if $m \in M \cup N$ does not contain a 1, $xm^\omega \in x\langle M^*\rangle \cup x\langle N^*\rangle$ does not contain 1's and it is a winning word. So the words in the set $x\langle(M \cup N)^*\rangle$ contain infinitely many 1's and they are losing. Moreover, since $x\langle K\rangle$ contains more than one 1, the words in $x(M \cup N)^*\langle K\rangle$ contain more than one 1 and they are all losing. So, the set $x\langle(M \cup N)^* K\rangle$ contains only losing words, hence a contradiction.

4. $\overline{W}$ is monotone. Suppose by contradiction that $\overline{W}$ is not monotone. Then there exist $x, y \in [k]^*$ and $M, N \subseteq [k]^*$ such that $xM < xN$ and $yN < yM$. So, xM and yN contain only losing words, xN and yM contain a winning word. If x contains more than one 1, all words in the first two sets are winning, hence a contradiction. If x contains one 1, then there is a word in N that does not contain 1's. Since yN contains only losing words, y contains more than one 1. So, all words in yM are losing, hence a contradiction. If x does not contain 1, then all words in M contain more than one 1, so all words in yM are losing, hence a contradiction.

□

5.5 Strong Selectivity

We proved in the previous section that strong monotonicity, strong concavity, do not constitute a characterization for half-positionality. Indeed, we observe that strong concavity is a stronger property than what is needed. It asks that no matter how two losing words are interwoven the results is still a losing word. The aim is to relate the alternation of player 0 between two behaviors to the switching between the two words. However, on a game graph, player 0 is not actually free to switch at every point between the two behaviors, but only at particular nodes where the two paths meet. Moreover, since a word should represent a positional behavior for player 0 on a finite graph, it should contain some periodicity. Both these requirements are achieved through the property of selectivity. However, selectivity can only be used in the hypothesis player 1 is using a positional strategy, since only in that case player 0 makes use of regular behaviors. In order to incorporate into selectivity, the ability to evaluate also non regular sets of words, we define a new stronger version of it.

Definition 5.5.1. *A goal W is* strongly selective *if and only if for all $x \in [k]^*$ and for all languages $M, N, K \subseteq [k]^*$ we have that $x\langle(M \cup N)^*K\rangle \leq x\langle M^*\rangle \cup x\langle N^*\rangle \cup x\langle K\rangle$.*

Selectivity and strong selectivity represent two weaker properties than strong concavity.

Lemma 5.5.1. *Every strongly concave goal is strongly selective.*

Proof. For all words $x \in [k]^*$, for all languages $M, N, K \subseteq [k]^*$, we have that $x\langle(M \cup N)^*K\rangle \subseteq x((\langle M^*\rangle \otimes_s \langle N^*\rangle) \otimes_s \langle K\rangle) \leq x\langle M^*\rangle \cup x\langle N^*\rangle \cup x\langle K\rangle$.

□

Unfortunately, the strong versions of selectivity and monotonicity proved not to be sufficient conditions to half positionality[2].

Lemma 5.5.2. *There exists a strongly monotone and strongly selective goal which is not half-positional.*

Proof. Let $k \in \mathbb{N}$, for all colors $i \in [k]$ and finite paths π, let $|\pi|_i$ be the number of edges colored by i on π, and let $|\pi|$ be the number of edges in π. Moreover for all $n \in \mathbb{N}$ let $\pi^{\leq n}$ be the prefix of length n of π. The strongly monotone and strongly selective goal is the set W of all the infinite words m such that, for all colors $i \in [k]$, the limit $\lim_{n\to+\infty} \frac{|m^{\leq n}|_i}{|m^{\leq n}|}$ exists and is finite. The goal is prefix independent. Indeed, let $\pi = x\pi'$ then for all $i \in [k]^*$ we have $\lim_{n\to+\infty} \frac{|\pi'^{\leq n}|_i}{|\pi'^{\leq n}|} = \lim_{n\to+\infty} \frac{|\pi^{\leq n+|x|}|_i - |x|_i}{|\pi^{\leq n+|x|}| - |x|} = \lim_{m\to+\infty} \frac{|\pi^{\leq m}|_i}{|\pi^{\leq m}|}$. The goal is also strongly selective. Indeed, suppose by contradiction that there exist a sequence $x \in [k]^*$, and three languages $M, N, K \subseteq [k]^*$ such that $x\langle(M \cup N)^*K\rangle$ contains one winning word and $x\langle M^*\rangle \cup x\langle N^*\rangle \cup x\langle K\rangle$ contains only losing words. In this case, M and N must be empty else any periodic word $\pi = m^\omega \in M^* \cup N^*$ with $m \in M \cup N$ has a finite limit $\lim_{n\to+\infty} \frac{|\pi^{\leq n}|_i}{|\pi^{\leq n}|} = \frac{|m|_i}{|m|}$, for all colors i. So, the set $\langle x(M \cup N)^*K\rangle = x\langle K\rangle$ and contains only losing words which is a contradiction. The above goal is not half-positional in the following arena $(\{u\}, \{v\}, u, \{(v, 0, u), (v, 1, u), (u, 0, v), (u, 1, v)\})$

[2]We thank Zielonka and Gimbert for pointing out the counterexample

with $k = 1$ (Fig 5.1(c)). Player 0 can win with a strategy with memory by choosing from V' to V the opposite of the color that player 1 chose from V to V' right before, thus yielding a path in $[k]^*(10)^\omega$ which has limit $\frac{1}{2}$ for both colors. However if player 0 uses a positional strategy, it will only choose one color from V' to V, let suppose without loss of generality that he chooses color 0. The player 1 can force a path $\pi = \prod_{i=0}^{+\infty}(00)^{2^i}(10)^{2^i}$. Then we have $|\prod_{i=0}^{l}(00)^{2^i}(10)^{2^i}| = \sum_{i=0}^{l} 4 \cdot 2^i = 4(2^{l+1} - 1)$, and $|(\prod_{i=0}^{l-1}(00)^{2^i}(10)^{2^i}) \cdot (00)^{2^l}| = 4(2^l + 2^{l-1} - 1)$. Moreover, $|\prod_{i=0}^{l}(00)^{2^i}(10)^{2^i}|_1 = \sum_{i=0}^{l} \cdot 2^i = (2^{l+1} - 1)$, and $|(\prod_{i=0}^{l-1}(00)^{2^i}(10)^{2^i}) \cdot (00)^{2^l}|_1 = \sum_{i=0}^{l-1} \cdot 2^i = 2^l - 1$. So we have $\frac{|\prod_{i=0}^{l}(00)^{2^i}(10)^{2^i}|_1}{|\prod_{i=0}^{l}(00)^{2^i}(10)^{2^i}|} = \frac{1}{4}$, moreover $\frac{|(\prod_{i=0}^{l-1}(00)^{2^i}(10)^{2^i})\cdot(00)^{2^l}|_1}{|(\prod_{i=0}^{l-1}(00)^{2^i}(10)^{2^i})\cdot(00)^{2^l}|} = \frac{2^l-1}{3(2^l-1)+2(2^l)} = \frac{2^l-1}{5(2^l)-3} = \frac{2^l-\frac{3}{5}}{5(2^l)-3} - \frac{\frac{2}{5}}{5(2^l)-3} < \frac{1}{5}$. This shows that in the limit $\frac{|\pi^{\leq n}|_1}{|\pi^{\leq n}|}$ oscillates between $\frac{1}{4}$ and something less than $\frac{1}{5}$.

□

Although the following theorem is obtained easily from the techniques developed in [GZ05], we think that it is worth mentioning that half positionality on arenas controlled only by player 0 is equivalent to the selectivity of the goal. Since the selectivity is similar in a way to strong concavity, we show that strong concavity is a condition useful to assert that, on decisions independent from player 1, player 0 prefers a fixed behavior rather than switching between two different ones. We prove the above statement by making use of the following lemma proved in [GZ05].

Lemma 5.5.3 ([GZ05])**.** *Let A be a finite co-accessible* [3] *automaton recognizing a language $L \subset [k]^*$ and having starting state q. Then, $\langle L \rangle$ is the set of infinite color sequences on the graph of A starting in q.*

Theorem 5.5.1. *A goal is selective if and only if it is half-positional on all arenas controlled by player* 0*.*

Proof. [only if] Suppose that a goal W is half-positional on all game graph controlled by player 0 but non-selective. Let $x \in [k]^*$ and $M, N, K \subseteq [k]^*$ be three recognizable languages such that $x\langle (M \cup N)^* K \rangle \not\leq x\langle M^* \rangle \cup x\langle N^* \rangle \cup x\langle K \rangle$. This means that there is a winning word in $x\langle (M \cup N)^* K \rangle$ and $x\langle M^* \rangle \cup x\langle N^* \rangle \cup x\langle K \rangle$ contains only losing words. Let G_x, G_M, G_N be the minimized finite automata recognizing the languages $\{x\}, M, N$, respectively, and having only one starting state with no transition returning to it and one final state with no transition exiting from it. Let G_K be the minimized finite automaton recognizing the language K, having only one starting state with no transition returning to it. We construct the game graph G by combining together the graphs G_x, G_M, G_N, G_K. Precisely we glue together the final state of G_x, the initial and final states of G_M and G_N and the initial state of G_K in a new node t. Observe that, by gluing together the initial and final states, the automata G_M, G_N recognize M^* and N^*, respectively. The initial state of G is the starting state of G_x. Thus the graph G recognizes the language $x(M \cup N)^* K$. Hence by Lemma 5.5.3, every infinite path in G is in $\langle x(M \cup N)^* K \rangle = x\langle (M \cup N)^* K \rangle$. Since this set contains a winning word, there is a winning strategy for player 0. However, if player 0 uses a positional strategy he cannot win. Indeed, player 0 reaches first the node t by constructing the

[3] An automaton is co-accessible if and only if from every state there is a path reaching an accepting state. It's easy to see that a minimized automaton is co-accessible.

color sequence x on G_x. In the node t player 0 chooses once and for all which of the subgraphs G_M, G_N, G_K he will use, so the infinite play will be of the form xm where m is an infinite path in G_M,G_N or G_K. By Lemma 5.5.3, $xm \in x\langle M^* \rangle \cup x\langle N^* \rangle \cup x\langle K \rangle$. But this set contains only losing words. Hence, xm is losing.

[if] Suppose that a goal W is selective, we prove by induction on the number of edges exiting from the nodes of the arena G controlled by player 0 that if there exists a winning strategy for player 0 then there exists a positional one. As base case there exists only one edge exiting from the nodes of G, hence player 0 has only one strategy, which is trivially positional. Suppose that in the arena there are n edges exiting from nodes of player 0 and that for all graphs with at most $n-1$ edges exiting from nodes of player 0, if player 0 has a winning strategy he has a positional one. Let t be a node of player 0 in G such that there is more than one edge exiting from t. We can partition the set of edges exiting from t in two disjoint non-empty sets E_α and E_β. Let G_α and G_β be the two subgraphs obtained from G by removing the edges of E_β and E_α, respectively. There are two cases to discuss. First, suppose that either in G_α or G_β player 0 has a winning strategy. Then, by inductive hypothesis he has a positional winning strategy. It is easy to see that such a strategy is winning in G too, indeed player 0 is able to play always in G_α or G_β since he controls every node.

Suppose now that player 0 has no winning strategy in G_α and in G_β. We prove the thesis by showing that player 0 has no winning strategy in G. Let M_α and M_β be the sets of all finite color sequences from t to t and K_α and K_β be the sets of all finite color sequences starting from t, in G_α and G_β, respectively. Such sets are regular languages: M_α and M_β are recognized by the automata having respectively G_α and G_β as state graphs, with starting node t and accepting set $\{t\}$. The sets K_α and K_β are the languages accepted by the automata with state graphs G_α and G_β, respectively, with starting node t and accepting set given by all the states.

Suppose now by contradiction that there exists a winning strategy for player 0 in G. Then this strategy will form a winning path π. Such a path cannot be in G_α or G_β, or else player 0 has a winning strategy in one of those subgraphs. So the path is in G and passes through t. Let x be the shortest prefix of π ending in t, then π belongs to the set $x\langle (M_\alpha \cup M_\beta)^*(K_\alpha \cup K_\beta)\rangle$, since it starts with x, then either loops forever from t to t in G_α and G_β, or possibly ends with an infinite path that never comes back to t. However, for $\gamma \in \{\alpha, \beta\}$, the sets $x\langle M_\gamma^* \rangle$ and $x\langle K_\gamma \rangle$ contain only paths in G_γ, so they are losing. Thus, we have $x\langle (M \cup N)^* K \rangle \not\leq x\langle M^* \rangle \cup x\langle N^* \rangle \cup x\langle K \rangle$, which contradicts selectivity.

□

Figure 5.5 summarizes the relations between half-positionality and the properties described in this chapter.

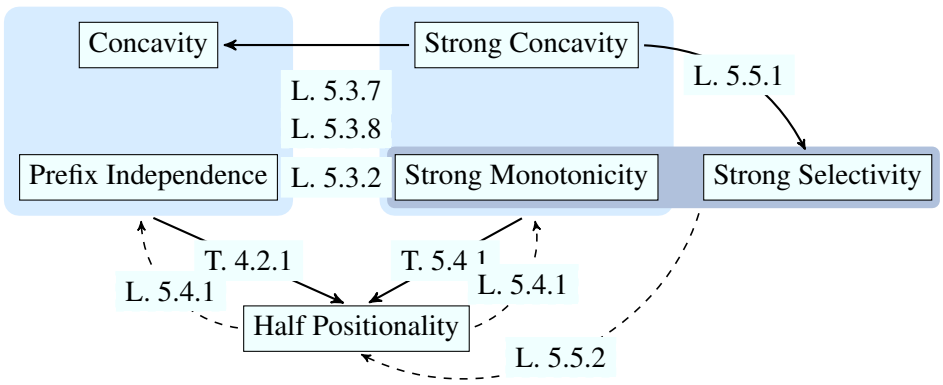


Figure 5.2: Summary of results. Continuous arrows represent a holding implication and dashed ones a false one. Arrows are labeled with the corresponding lemma or theorem. Moreover, a gray box represents a conjunction of conditions.

Conclusion

In this thesis we discussed how to determine scheduling plans for ensuring that on a system's executions every task is performed with a desired asymptotic frequency or uniform asymptotic frequency. We showed that when the scheduler has complete control of the system it is possible to determine a suitable plan in polynomial time, and that when the environment can influence the executions, determining the existence of a scheduling plan is a $Co - NP$-complete problem. In the latter case, we also proposed an exponential space algorithm for the computation of winning strategy for the scheduler.

All the above results hold for k-colored graphs and can be used when the tasks can be decomposed into atomic components representable by means of a single edge. Although time and length are considered to belong to a dense domain, we can still consider approximations to multiple of a given unit of costs. Such a unit becomes the value of an atomic task. However, every time we decompose a task in its components, we add to the graphs as many consecutive edges and a nodes in as the number of atomic components. In non-preemptive situation where a task needs to complete before a new one is allowed, this is not a problem. Indeed, every task is represented as a sequence of consecutive edges whose nodes just allow one exiting edge representing the next atomic task to be performed. In this case our algorithms can be easily modified to avoid a complexity explosion. For the deterministic scheduling we can just have one variable for a sequence of consecutive edges in the feasibility systems. Moreover, the algorithm for the computation of a winning strategy is exponential in the number of nodes of player 1, and nodes with just an exiting edge can be safely attribute to player 0. On the other hand, when the scheduling is preemptive, we are in the hypothesis that tasks can he halted only at some discrete instants in time in order to decompose it in atomic steps. So, the fact that every intermediate atomic activity can be followed by an activity from another task, induces a necessary complexity explosion in our algorithm. The explosion becomes worse the more refined the unit of cost becomes. Hence, it may be worth investigating, the existence of frequency-f paths on dense-cost models, such as timed graphs.

Bibliography

[ACD90] Rajeev Alur, Costas Courcoubetis, and David L. Dill. Model-checking for real-time systems. In *LICS*, pages 414–425, 1990.

[AFK87] K. R. Apt, N. Francez, and S. Katz. Appraising fairness for languages in distributed programming. In *POPL*, pages 136–145, 1987.

[AH98] R. Alur and T.A. Henzinger. Finitary fairness. *ACM Trans. on Programming Languages and Systems*, 20(6), 1998.

[AHK02] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-Time Temporal Logic. *Journal of the ACM*, 49(5):672–713, 2002.

[BFMM09] A. Bianco, M. Faella, F. Mogavero, and A. Murano. Balanced Paths in Colored Graphs. In *MFCS'09*, volume 5734 of *LNCS*, pages 149–161. Springer, 2009.

[BFMM10a] A. Bianco, M. Faella, F. Mogavero, and A. Murano. Quantitative fairness games. In *QAPL10, 8th Workshop on Quantitative Aspects of Programming Languages*, Electronic Proceedings in Theoretical Computer Science, 2010. To appear.

[BFMM10b] Alessandro Bianco, Marco Faella, Fabio Mogavero, and Aniello Murano. Exploring the boundary of half positionality. In *CLIMA XI*, pages 171–185, 2010.

[Bry01] R. Bryant. *Borel Determinancy and Metamathematic*. PhD thesis, University of North Texas, 2001.

[CD73] E.G. Coffman and P.J. Denning. *Operating Systems Theory*. Prentice Hall, 1973.

[CLRS01] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2001. Second Edition.

[dA99] L. de Alfaro. From fairness to chance. *ENTCS*, 22:55–87, 1999.

[EMSS90] E. Allen Emerson, Aloysius K. Mok, A. Prasad Sistla, and Jai Srinivasan. Quantitative temporal reasoning. In *CAV*, pages 136–145, 1990.

[Fra86] N. Francez. *Fairness*. Springer, 1986.

[GZ05] H. Gimbert and W. Zielonka. Games where you can play optimally without any memory. In *CONCUR'05*, volume 3653 of *LNCS*, pages 428–442. Springer, 2005.

[HL03] F. S. Hillier and G. J. Lieberman. *Introduction to Operations Research*. MCGrawHill, 2003.

[Kop06] E. Kopczyński. Half-positional determinancy of infinite games. In *ICALP'06*, volume 4052 of *LNCS*, pages 336–347. Springer, 2006.

BIBLIOGRAPHY

[Kup95] O. Kupferman. *Model Checking for Branching-Time Temporal Logics*. PhD thesis, The Technion, 1995.

[KVW01] O. Kupferman, M.Y. Vardi, and P. Wolper. Module checking. *Information and Computation*, 164:322–344, 2001.

[Kwi89] M. Kwiatkowska. Survey of fairness notions. In *Information and Software Technology*, volume 31(7), pages 371–386, 1989.

[LCST98] J. C. S. Lui, M. F. Chan, O. K. Y. So, and T. S. Tam. Balancing workload and communication cost for a distributed virtual environment. In *Advances in Multimedia Information Systems*, volume 1508/1998, pages 130–135. Lecture Notes in Computer Science, 1998.

[LPS81] D. Lehmann, A. Pnueli, and J. Stavi. Impartiality, justice and fairness: The ethics of concurrent termination. In *ICALP'81*, volume 115 of *LNCS*, pages 264–277. Springer, 1981.

[Mar75] A.D. Martin. Borel determinancy. 102(2):363–371, 1975.

[MP91] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer, 1991.

[NW88] G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization*. Wiley-Interscience, 1988.

[Phi03] R. Philippe. *Stochastic networks and queues*. Springer, 2003.

[Pin08] M. L. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Springer, 2008.

[Sch86] A. Schrijver. *Theory of linear and integer programming*. John Wiley and Sons, 1986.

[SG98] A. Silberschatz and P. B. Galvini. *Operating System Concepts*. Addison Wesley, 1998.

[SL85] K. E. Stecke and T. L.Morin. The optimality of balancing workloads in certain types of flexible manufacturing systems. In *European Journal of Operational Research*, pages 68–82, 1985.

[Tan03] A. Tanenbaum. *Computer Networks*. Pearson Prentice, 2003.

[VVK05] Hagen Völzer, Daniele Varacca, and Ekkart Kindler. Defining fairness. In *CONCUR*, pages 458–472, 2005.

List of Figures