

Contents

Introduction	1
1 Interactions networks	3
1.1 Sequence and network data	3
1.2 Elements of graph theory	4
1.3 Combinatorial optimization	4
2 State of the art	7
2.1 Gene selection	7
2.2 (Protein-protein?) Interaction networks	8
2.3 The Maximum-Weight Connected Subgraph problem	8
3 xHeinz: a cross-species module discovery tool	13
3.1 Assigning weights	13
3.2 Algorithmic problem	13
3.3 Results	16
4 Tight hardness bounds for the MWCCS problem	17
4.1 APX-hardness of the MWCCS problem	18
4.2 Polynomial-time cases for the MWCCS problem	24
5 The Ratio-Bounded MWCS	29
5.1 A more general variant of the Budget-Constrained MWCS	29
5.2 Pseudo-polynomial algorithms from MWCS	31
Conclusion	33
Bibliography	a

Introduction

- genomic, transcriptomic
- sequencing
- database construction (networks ?)
- bioinformatics pipeline
- type of data
- integrative approaches

This thesis is structured in three main parts. In ?? we present an outline of the domain, first with a general overview, and later with an analysis of the state of the art. Chapter 1 is separated in two main sections, introducing some biology notions and some computer science background relating to this work. Chapter 2 presents a thorough review of the existing work and main methods that either serve as direct support or are relevant for our work for both module discovery and complexity analysis of similar problems.

Section 2.3.5 is dedicated to our methodological and software contribution. It is mainly constituted of chapter 3, which is sectionned as follows. Section 3.1 establishes the basis for module discovery, assigning weights to genes based on experimental evidence. This chapter describes two methods for weight assignment, from transcriptomic expression profiles, and from secondary evidence obtained through proteomic analysis. Section 3.2 introduces an important algorithmic contribution to the module detection problem by describing a robust mathematical model, serving as our definition of cross-species module, with flexible requirement of sequence conservation. Multiple techniques could be used to express and resolve this model, which implicitly includes the Maximum-Weight Connected Subgraph problem. It will first be expressed as a Mixed-Integer Linear program, and we will present an in-depth algorithmic implementation to solve the problem in the general case. Section 3.3 presents our biological results and empirical analysis of algorithmic performance.

Finally, section 3.3 is dedicated to the algorithmic complexity analysis of the combinatorial optimization problem underlying our model. Chapter 4 provides detailed proofs for the frontier of difficulty for the problem introduced by our model, producing a precise separation of complexity classes between types of input graphs. Chapter 5 describes a general scheme to construct efficient, pseudo-polynomial algorithms for a subclass of our initial problem.

Chapter 1

Interactions networks

1.1 Sequence and network data

1.1.1 Genome, transcriptome, proteome

1.1.2 Modeling the living, biological networks

Biological networks are abstract representations of biological entities interconnected by some criteria. They can represent for example the relationships between species inside an ecosystem, or interconnections between cell types in any multicellular organism.

In this work, we are mostly interested in networks at the biomolecular level. Many such networks exists, to name a few:

- *Metabolic networks* represent biochemical reactions between substrates, enzymes and metabolites, and cluster them into pathways,
- *Gene co-expression networks* represent the similarity of expression between genes in some biological setup, by interconnecting pairs of genes simultaneously expressed,
- *Gene regulatory network* represent the indirect regulatory actions of genes, from proteins and transcription factors to gene expression levels,
- *Protein-protein interaction networks* represent interactions between two proteins, usually of the same species.

On the one hand biological networks can be seen as observed or inferred facts, where the network represents the knowledge ; e.g. a known pathway that connect chemical reactants and products through enzymes. On the other hand they can be seen as an abstract representation of knowledge where nodes represent entities and edges represent some form of deduced connections ; XXX e.g. a gene co-expression network which can be constructed from the control and condition expression profiles of the genes, with a statistical inference over the two samples resulting in the presence or absence of the edges. (trop embrouille – Macha) XXX

Let us stress the importance of biological networks in modern biology. They structure our understanding of biological systems in such ways that both allow a comprehension of biological processes at the system level, and permit automated processing of the knowledge that they represent. As automated processing enabling tools, they can serve as both knowledge bases for local decisions and as global networks that can serve as XXX substrate (de quoi parles-tu? – Macha) XXX for integrated analysis.

XXX.

The most fitting abstraction for those biological networks are discrete mathematics' graphs (XXX to be formally defined in ...XXX).

Protein-protein interactions (PPI) networks play an important role in this work, and we will present them in more detail.

Protein-Protein Interactions

1.1.3 Gene expression

Measuring gene expression levels

Differential analysis

- better understanding of cellular processes
- biomarkers discovery

1.2 Elements of graph theory

1.2.1 Graphs

Let us recall some basic material related to graphs. A graph $G = (V, E)$ consists of a set of vertices V and a set of edges (unordered pairs of vertices) E . We say that G is node-weighted if a function $w: V \rightarrow \mathbb{R}$ is provided. Given a graph $G = (V, E)$, its subgraph $G' = (V', E')$ is said to be *induced* if G' has exactly the edges that appear in G over the vertex set $V' \subseteq V$, that is $E' = \{(x, y) \in E \mid x, y \in V'\}$. We denote the graph *induced* by the node set V' in G by $G[V']$.

1.3 Combinatorial optimization

Dynamic programming

Decision trees, Branch and bound, Branch and cut

Linear programming, Mixed integer linear programming

1.3.1 Complexity

APX-hardness One well studied APX-hard problem is the $\text{MAX-3SAT}(B)$ problem and is defined by Papadimitriou and Yannakakis (1991) as follows. Given a collection $C_q = \{c_1, \dots, c_q\}$ of q clauses where each clause consists of a set of three literals over a finite set of n boolean variables $V_n = \{x_1, \dots, x_n\}$ and every literal occurs in at most B clauses, is there a truth assignment of V_n satisfying the largest number of clauses of C_q ?

Pseudo-polynomial time

Chapter 2

State of the art

2.1 Gene selection

2.1.1 Gene expression

In the last two decades, the large adoption of *microarray* technologies have dramatically changed the landscape of biology. Microarray technologies are high-throughput screening methods for biological material, that allow experimenters to assay the amount the quantity of a specific material on a large scale. Where previous assay methods were rate limited, non-parallelizable and non-multiplexable, microarrays allow multiple material to be screened for and quantified in parallel. Those methods also usually involved human interaction and were expensive, whereas microarray are typically the size of a microscope slide, are more sensitive and can be automated. This ability to quantify many molecular markers in parallel led to a whole new level of understanding of complex biological processes.

There exists many different kinds of microarray, for different biological material. Amino-acid sequences microarray are an important one, and the first to be introduced. T.-W. Chang (1983) realised that he could create a bidimensional array of sequence probes that would allow him to screen for antibodies by assessing presence and quantity of bound material. He later developed the concept in a series of patents (T. Chang 1986, 1989, 1992) which led to a whole new industry.

But by far the most important kind of microarray are the one that screen DNA sequences. They evolved from *DNA blotting*¹ techniques introduced by Southern (1975), where DNA fragments are first separated then selectively hybridized by a probe. Two decades ago, Schena et al. (1995) used robotic printing to prepare microarrays plates with fluorescently labeled complementary DNA (*cDNA*) sequences from *Arabidopsis thaliana*. Using high-precision laser to excite the fluorescent markers, they used those microarrays for quantitative expression measurement of the corresponding genes *mRNA* expression levels.

¹Also known as *Southern blotting*.

2.1.2 Gene set selection

One of the key concepts to understand biological processes is that of *modules* within biological networks. Modules are considered to be sets of entities (genes, proteins, etc.) that function in a coordinated fashion or physically interact (for a review see Mitra et al. (2013)).

The problem of finding gene modules within a biological network was first solved using simulated annealing by Ideker et al. (2002).

- context, differential analysis
- traditionnally, gene centric (cf. next subsection)
- First: Golub et al. 1999

2.1.3 Cross-species discovery

- single gene conservation: Noort et al. 2003

2.2 (Protein-protein?) Interaction networks

Increasingly advanced experimental methods are used to provide evidence of existing interactions and nowadays comprehensive resources provide access to this knowledge XXX.

- String: (Szklarczyk et al. 2014)

2.3 The Maximum-Weight Connected Subgraph problem

In section 2.3.1 we first introduce the STEINER TREE, the PRIZE-COLLECTING STEINER TREE (PCST), and the MAXIMUM-WEIGHT CONNECTED SUBGRAPH (MWCS) problems. We also provide the main complexity results for the MWCS problem, and since a number of those results mostly follows from results for the PCST problem, we will provide the most important ones for this problem too. In section 2.3.3 we then describe in more depth their use in biological modeling contexts through integrative approach to gene set selection.

2.3.1 Problems introductions

The STEINER TREE problem is an extremely well known combinatorial optimization problem, which is part of Karp's original 21 NP-complete problems (Karp 1972). It has its origin in the geometry of Jakob Steiner's eponym Steiner problem, and pertain to the class of mathematical optimization problems over graphs. A large number of variations of this problem exists: the *Steiner tree problems*. Hauptmann and Karpinski (2014) maintain an extensive and up-to-date compendium of those variants.

The MAXIMUM-WEIGHT CONNECTED SUBGRAPH problem, or MWCS problem, falls inside the same classification as the various Steiner tree problems: its a combinatorial optimization over graphs problem. It is informally defined as follows.

Given a graph and a real-valued weight for each vertex, the goal is to find the connected set of vertices that maximizes the sum of the weights.

Note that the solution is trivial, full or empty, if the vertices' weights are respectively all positive or all negative.

From a computer science point of view, the MWCS problem is simple in its definition. Nevertheless, it is actually a very difficult problem to solve, and in many cases remains intractable. The first text to prove NP-hardness of the problem is the unpublished manuscript from Vergis (1983). Manuscript that Johnson (1985, Section 5) acknowledge when he looked into a series of graph problems in his famous *NP-Completeness Columns*. The proof is based on the reduction from the STEINER TREE problem, provided by Garey and Johnson (1979). Johnson made the fundamental observation that the solution to the MWCS problem can always be reduced to a tree, since the additional edges serve no purpose. Karp later provided another proof of NP-hardness for the MWCS problem in (Ideker et al. 2002, Supplementary Material), by providing a reduction from the MINIMUM SET COVER problem, another problem which is one of his first 21 NP-complete problems (Karp 1972).

One of the variants of the STEINER TREE problem is the PRIZE-COLLECTING STEINER TREE problem, or PCST problem. It is an *utility versus cost optimization*² variant of the STEINER TREE problem, for which the first proof of NP-hardness is in (Camerini et al. 1979). Even if the result is easy³, Feigenbaum et al. (2001) were the firsts to prove that the PCST problem is NP-hard to approximate within any constant ratio $0 < \epsilon < 1$, or APX-hard, using a reduction from SAT. This variant is highly relevant to our context as there exists bidirectional reductions between the PCST and MWCS problems, and for a long time reducing an MWCS instance to a PCST instance was the technique of choice to actually solve the problem.

And indeed, lately Álvarez-Miranda et al. (2013a) recognized that the MWCS problem is actually APX-hard itself. Using the SAT reduction previously mentioned (Feigenbaum et al. 2001), they extended the result to MWCS using a straightforward reduction of PCST to MWCS.

In the same way the STEINER TREE broadly defines a large set of related problems, the MWCS problem defines itself a set of closely related problems. Nowadays those problems have a very wide breadth of applications, of which: social network sciences, operations research, certainly networks design, and system biology, which is our main interest in this manuscript.

Based on the basic version of the MWCS problem, the principal variants are the *constrained* versions, with an allocated budget and an additional costs assigned to each vertex. Of which the k -CARDINALITY MWCS, the CARDINALITY-CONSTRAINED MWCS, and the BUDGET-CONSTRAINED MWCS serve interesting purposes in our context. The first one require that the solution be comprised of exactly k vertices, whereas the second variant require that the solution includes at most K vertices. The last variant assigns an additional positive cost to each vertex, and requires that the sum of the costs be at most a given budget B . Clearly, assigning a positive cost of 1 for each node of the CARDINALITY-CONSTRAINED

²As defined by Conrad et al. (2007).

³According to Feigenbaum et al. (2001, footnote 12).

MWCS problem provides a trivial reduction to the BUDGET-CONSTRAINED MWCS problem.

Note that for all those constrained versions, the problems remain non-trivial even when the nodes' weights are all positives. Furthermore, note that while removing the connectivity constraint in the basic version of the problem makes it trivial, in those variants the problems then become equivalent to the 0-1 KNAPSACK problem (which is another one of Karp's original 21 NP-complete problems (Karp 1972)).

They can all be defined either on graphs or on directed graphs, and there exists rooted variants where one (or multiple) root(s) have to be selected in the solution.

There exist minimization variants for all those problems, which are strictly equivalent from an optimality standpoint⁴. XXX they might differ from approximation standpoint though (working draft –*Approximation algorithms for finding a k -connected subgraph of a graph with minimum weight*, Tamás Hajba–). Need to read in more depth XXX.

XXX Variants with costs on the edges also exists. Do we talk about them ? Never used afterward XXX

2.3.2 Problems introductions

Hochbaum and Pathria (1994) first described the fixed cardinality variant of the problem (k -CARDINALITY MWCS). They relate its use in two contexts. First in off-shore oil-drilling where each facility is represented by a node and the weight their costs over benefits. Second in forest harvesting where we need to find the k connected parcel to harvest considering their associated benefits. They call it the CONNECTED k -SUBGRAPH problem. They were the first to observe that, for this variant, since adding a constant to the weight of each node does not change the optimal set of nodes, in the optimality context the nodes' weights can all be non-negative. They also show that the problem is NP-hard even for bipartite or planar input graphs or if the nodes' weights are boolean.

Lee and Dooley (1998) reintroduced the k -CARDINALITY MWCS problem, and called it simply the MAXIMUM-WEIGHT CONNECTED GRAPH (MCG) problem. They were the firsts to introduce the rooted variant where a single root is provided for the solution, that they called the CONSTRAINED MCG (CMCG). They used this rooted variant to provide a decomposition scheme of the MWCS problem into multiple CMCG subproblems. They acknowledge that the optimal solution is NP-hard to find, and provide heuristics for their incremental roots selection.

Gomes' team recently looked into the budget constrained version of the problem, both rooted and unrooted variants, that they apply in the context of *conservation planning* (Conrad et al. 2007; Gomes et al. 2008; Dilkina and Gomes 2010).

2.3.3 Biological modeling with MWCS

Biological context: Dittrich et al. 2008, Backes et al. 2012

⁴It is equivalent to minimizing the opposite or the inverse of the nodes' weights

Yamamoto et al. 2009 introduce an heuristic method based on betweenness centrality measures in order to converge faster to optimality. They further use the cardinality-constrained MWC S variant to detect core components in gene interaction networks.

2.3.4 Module as a connected cluster of genes

A possible formulation for the problem of finding modules within a network is to look for connected sub-networks that maximize weights on the nodes. These weights typically represent some measure of biological activity, for example the expression level of genes. The seminal work of Ideker et al. (2002) XXX shows it XXX.

Finding the optimal (with respect to sum of weights) module in a biological network has been formally defined as the MAXIMUM (NODE-)WEIGHT CONNECTED SUBGRAPH problem (MWC S) Dittrich et al. 2008.

XXX

- Heuristic: Ideker et al. 2002, Mitra et al. 2013, ...
- Exact: Dittrich et al. 2008

2.3.5 Solving the MWC S problems

Through the Prize-Collecting Steiner Tree problem

Through a direct method (Miranda-Alvarez)

- rooted: Álvarez-Miranda et al. 2013b
- unrooted: Álvarez-Miranda et al. 2013a

Chapter 3

xHeinz: a cross-species module discovery tool

3.1 Assigning weights

3.1.1 Peptides mappings

3.1.2 Gene weight through expression profiles

3.2 Algorithmic problem

3.2.1 Mathematical model

We consider the conserved active modules problem in the context of two species networks, which we denote by $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Nodes in these networks are labeled by their activity—defined by $w \in \mathbb{R}^{V_1 \cup V_2}$ and conserved node pairs are given by the symmetric relation $R \subseteq V_1 \times V_2$. The aim is to identify two maximal-scoring connected subnetworks, one in each network, such that a given fraction α of module nodes are conserved. The formal problem statement is as follows:

Problem 1 (Conserved active modules). *Given $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$, $w \in \mathbb{R}^{V_1 \cup V_2}$ and $R \subseteq V_1 \times V_2$, the task is to find a subset of nodes $V^* = V_1^* \cup V_2^*$ with $V_1^* \subseteq V_1$ and $V_2^* \subseteq V_2$ such that the following properties hold.*

- **Activity:** Node activity scores are given by $w \in \mathbb{R}^{V_1 \cup V_2}$, where positive scores correspond to significant differential expression. For details see Section ???. We require that the sum $\sum_{v \in V^*} w_v$ is maximal.
- **Conservation:** Conserved node pairs are given by the relation $R \subseteq V_1 \times V_2$. We require that at least a certain fraction α of the nodes in the solution must be conserved, that is, $|U^*| \geq \alpha \cdot |V^*|$ where $U^* := \{u \in V_1^* \mid \exists v \in V_2^* : uv \in R\} \cup \{v \in V_2^* \mid \exists u \in V_1^* : uv \in R\}$.
- **Modularity:** We require that the induced subgraphs $G_1[V_1^*]$ and $G_2[V_2^*]$ are connected.

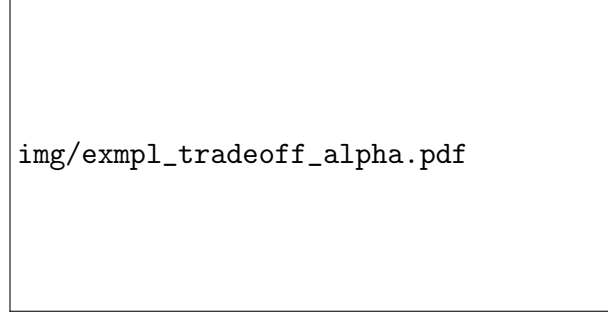


Figure 3.1: **Trade-off between activity and conservation.** Three optimal solutions (indicated in yellow) for varying conservation ratios α in a toy example instance. Node activities are given next to the nodes, conserved node pairs are linked by dotted lines. The activity of a conserved module is the sum of the activities of its comprising nodes. The parameter α denotes the minimum fraction of nodes in a solution that must be conserved, *i.e.* connected by a dotted line.

The model allows a trade-off between conservation and activity. If no conservation is enforced ($\alpha = 0$), the solution will correspond to two independent maximum-weight connected subgraphs. Conversely, if complete conservation is required ($\alpha = 1$), the solution can only consist of conserved nodes, which results in lower overall activity. The user controls this trade-off by varying the value of the parameter α from 0 to 1. The activity score monotonically decreases with increasing α —see Fig. 3.1.

Since the maximum-weight connected subgraph problem, which occurs as a subproblem for $\alpha = 0$, is NP-hard Johnson 1985, the problem of finding conserved active modules is NP-hard as well.

3.2.2 Mixed-Integer Linear program

We formulate the conserved active modules problem as an integer programming (IP) problem in the following way.

$$\max \sum_{v \in V_1 \cup V_2} w_v x_v \quad (3.1)$$

$$\text{s.t. } m_u = \max_{uv \in R} \{x_u x_v\} \quad u \in V_1 \quad (3.2)$$

$$m_v = \max_{uv \in R} \{x_u x_v\} \quad v \in V_2 \quad (3.3)$$

$$\sum_{v \in V_1 \cup V_2} m_v \geq \alpha \sum_{v \in V_1 \cup V_2} x_v \quad (3.4)$$

$$G_1[\mathbf{x}] \text{ and } G_2[\mathbf{x}] \text{ are connected} \quad (3.5)$$

$$x_v, m_v \in \{0, 1\} \quad v \in V_1 \cup V_2 \quad (3.6)$$

Variables $\mathbf{x} \in \{0, 1\}^{V_1 \cup V_2}$ encode the presence of nodes in the solution, *i.e.*, for all $v \in V_1 \cup V_2$ we want $x_v = 1$ if $v \in V^*$ and $x_v = 0$ otherwise. The objective function (3.1) uses

these variables to express the activity of the solution, which we aim to maximize. Variables $\mathbf{m} \in \{0, 1\}^{V_1 \cup V_2}$ encode the presence of conserved nodes in the solution. Constraints (3.2) encode that a node $u \in V_1$ that is present in the solution ($x_u = 1$) is conserved if there exists a related node $v \in V_2$ ($uv \in R$) that is also present in the solution ($x_v = 1$). Similarly, constraints (3.3) defines conserved nodes in V_2 that are present in the solution. The fraction of conserved nodes in the solution is at least α as captured by (3.4). In addition, we satisfy the modularity property by requiring in (3.5) that $G_1[\mathbf{x}]$ and $G_2[\mathbf{x}]$ are connected.

This formulation satisfies the properties of activity, conservation and modularity.

Activity. Variables $\mathbf{x} \in \{0, 1\}^{V_1 \cup V_2}$ encode the presence of nodes in the solution, *i.e.*, for all $v \in V_1 \cup V_2$ we want $x_v = 1$ if $v \in V^*$ and $x_v = 0$ otherwise. The objective function (3.1) uses these variables to express the activity of the solution, which we aim to maximize.

Conservation. Variables $\mathbf{m} \in \{0, 1\}^{V_1 \cup V_2}$ encode the presence of conserved nodes in the solution. Recall that a node $u \in V_1^*$ ($u \in V_2^*$) that is present in the solution is conserved if there is another node $v \in V_2^*$ ($v \in V_1^*$) in the solution such that the two nodes form a conserved node pair $uv \in R$ ($vu \in R$). This corresponds to constraints (3.2) and (3.3). We linearize $x_u x_v$, in a standard way, by introducing binary variables $\mathbf{z} \in \{0, 1\}^R$ such that $z_{uv} = x_u x_v$ for all $uv \in R$:

$$z_{uv} \leq x_u \quad uv \in R \quad (3.7)$$

$$z_{uv} \leq x_v \quad uv \in R \quad (3.8)$$

$$z_{uv} \geq x_u + x_v - 1 \quad uv \in R \quad (3.9)$$

$$z_{uv} \in \{0, 1\} \quad uv \in R \quad (3.10)$$

Subsequently, we model the max function in (3.2) and (3.3) as follows.

$$m_u \geq z_{uv} \quad uv \in R \quad (3.11)$$

$$m_v \geq z_{uv} \quad uv \in R \quad (3.12)$$

$$m_u \leq \sum_{uv \in R} z_{uv} \quad u \in V_1 \quad (3.13)$$

$$m_v \leq \sum_{uv \in R} z_{uv} \quad v \in V_2 \quad (3.14)$$

We model the required degree of conservation by constraint (3.4).

Modularity. Constraint (3.5) states that the nodes encoded in the solution \mathbf{x} induce a connected subgraph in both G_1 and G_2 . There are many ways to model connectivity, *e.g.*, using flows or cuts Magnanti and Wolsey 1995. Cut-based formulations perform better in practice Dilkina and Gomes 2010. Recently, (Álvarez-Miranda et al. 2013a) have introduced a cut-based formulation that only uses node variables. In an empirical study, the authors show that their formulation outperforms other cut-based formulations. We model connectivity along the same lines. Since the constraints that we will describe are similar for both

graphs, we introduce them only for graph $G_1 = (V_1, E_1)$.

$$\sum_{v \in V_1} y_v \leq 1 \quad (3.15)$$

$$y_v \leq x_v \quad v \in V_1 \quad (3.16)$$

$$x_v \leq \sum_{u \in \delta(S)} x_u + \sum_{u \in S} y_u \quad v \in V_1, \{v\} \subseteq S \subseteq V_1 \quad (3.17)$$

$$y_v \in \{0, 1\} \quad v \in V_1 \cup V_2 \quad (3.18)$$

where $\delta(S) = \{v \in V_1 \setminus S \mid \exists u \in S : uv \in E_1\}$ denotes the *neighbors* of S . The modularity property states that \mathbf{x} should induce a connected subgraph in G_1 . We model this by introducing binary variables $\mathbf{y} \in \{0, 1\}^{V_1}$ that determine the root node. Constraints (3.15) and (3.16) state that at most one node $v \in V_1^*$ is the root node--in which case $y_v = 1$. Constraints (3.17) state that x_v can only be 1 if for all sets $S \subseteq V$ containing v it holds that either the root node is in S or there is a neighbor u of S in the solution. There is an exponential number of such constraints. We therefore do not add all these constraints to our initial formulation. Instead, we use a branch-and-cut approach, that is, at every node of the branch-and-bound tree we identify all violated constraints and add them to the formulation. Finding violated inequalities corresponds to solving a minimum cut problem, which we do using the algorithm by (Boykov and Kolmogorov 2004).

To further improve the performance, we have strengthened our model with the following constraints.

$$y_v = 0 \quad v \in V, w_v < 0 \quad (3.19)$$

$$\sum_{u \in V} y_u \geq x_v \quad v \in V, w_v \geq 0 \quad (3.20)$$

$$y_v \leq 1 - x_u \quad u, v \in V, u < v, w_u \geq 0, w_v \geq 0 \quad (3.21)$$

$$x_v \leq \sum_{u \in \delta(\{v\})} x_u + y_v \quad v \in V \quad (3.22)$$

As an optimization, we require that the root node must be a non-negatively weighted node in (3.19). Constraints (3.20) state that if a non-negatively weighted node v is present in the solution then there must be a root node. Constraints (3.21) are symmetry breaking constraints, they require that among all non-negatively weighted nodes in the solution, the root node is the smallest one--according to some arbitrary order. Finally, constraints (3.22) correspond to the cases where the set S in (3.17) is a singleton.

3.3 Results

Chapter 4

Tight hardness bounds for the MWCCS problem

In this chapter, we will outline the frontier of complexity that characterizes the MAXIMUM-WEIGHT CROSS-CONNECTED SUBGRAPH (MWCCS) problem. We will demonstrate that the bipartite relationship plays a major role in the separation between complexity classes, as do the types of input graphs. Furthermore, we will provide constructive proofs, with algorithmic solution that efficiently solve some instances of the problem in polynomial time.

The difficulty of the problem is not only dependent on the characteristics of the two main input graphs, but also of the relationship between the nodes of those two graphs. In Hume et al. 2015 we suggested that this characteristic is as important as those of the two graphs in defining the frontier of difficulty. We hypothesised that the problem might be polynomial-time solvable in some instances with a simpler relationship function.

In section 4.1 we demonstrate that the MWCCS problem is inapproximable¹ up to any arbitrary factor $\sigma < 1.0014$, it is thus an APX-hard problem. Exactly, we show that the APX-hardness holds for all inputs complex enough to represent the following cases:

1. one of the two graphs is a *binary caterpillar tree*, the other a *binary tree*, and the bipartite relationship is *injective*,
2. one of the two graphs is a *binary tree*, the other a *general graph*, and the bipartite relationship is *bijective*.

The proofs are respectively in section 4.1.1 and section 4.1.2.

Section 4.2 is separated into two main subsections dedicated to polynomially solvable cases of the MWCCS, both using constructive algorithmic description and with sub-problem definition and reduction. XXX In Hume et al. 2015 we suggested that the case where both graphs are *trees* and the relationship is *bijective* might be polynomial-time solvable. (qu'est-ce que cela veut dire ? – Macha) XXX We will give a proof for this conjecture in section 4.2.1. This is important in that it draws a clear complexity frontier, distinctly dependent on the

¹NP-hard to approximate.

bipartite relationship. Finally, in section 4.2.2 we provide a general algorithm to solve MWCCS. Doing so we introduce a new problem, RB-MWCS, to which chapter 5 is dedicated. Given that one of the two graphs is polynomially enumerable, we provide an efficient reduction from an MWCCS instance to a polynomial number of RB-MWCS instances. In this situation, MWCCS is thus as difficult as RB-MWCS: it is solvable in polynomial time when the second graph has a bounded treewidth².

The exploration of the frontier of complexity is done in a similar fashion in the two contexts. Adding constraints on one hand leads to a relaxation of some other on the other hand, in order to stay within the same category of difficulty. In the hardness proof context, changing from an injective to a bijective mapping function requires that one of the two trees becomes a general graph for the problem to stay difficult. In the polynomial-time algorithms context, requiring that one of the graph that was previously a binary tree becomes more general requires that the other becomes polynomially enumerable to stay solvable³.

4.1 APX-hardness of the MWCCS problem

In this section we prove the inapproximability of two specific cases of the MWCCS problem. First, we prove that if the mapping between G_1 and G_2 is an injective function, if G_1 is a binary caterpillar tree, and if G_2 is a binary tree, MWCCS is APX-hard and can not be approximated within factor 1.0014. Then, we prove that if the mapping is a bijective function, the problem is as hard to approximate as when considering a binary tree and a graph. These results in themselves shade some light on the role of the relationship function with respect to the difficulty of the problem.

Both proofs consist in L-reductions from the APX-hard MAX-3SAT(B) problem (cf. section 1.3.1).

4.1.1 Injective relationship function, binary trees

Proposition 1. *The MWCCS problem for a binary caterpillar tree and a binary tree is APX-hard and not approximable within factor 1.0014 even when the mapping M is an injective function and a complete conservation (i.e. $\alpha = 1$) is required.*

We first describe how we build an instance of MWCCS corresponding to an instance of MAX-3SAT(B). Given any instance (C_q, V_n) of MAX-3SAT(B), we build a binary caterpillar tree $G_1 = (V_1, E_1)$ with weight function w_1 , a binary tree $G_2 = (V_2, E_2)$ with weight function w_2 , and a mapping M as follows.

The binary caterpillar graph G_1 is defined as follows. The vertex set is $V_1 = \{r, l_i, c_j, dl_i, dc_j \mid 1 \leq i \leq n, 1 \leq j \leq q\}$. The edge set is given by the following equation.

$$E_1 = \{(c_j, dc_j), (l_i, dl_i) \mid 1 \leq i \leq n, 1 \leq j \leq q\} \cup \{(dc_q, r), (r, dl_1)\}$$

²cf. section 4.2.2 for the problem reduction and chapter 5 for the RB-MWCS hardness analysis.

³In polynomial time.

$$\{(dc_j, dc_{j+1}), (dl_i, dl_{i+1}) \mid 1 \leq i < n, 1 \leq j < q\}.$$

The weight function w_1 is defined as follows: for all $1 \leq i \leq n$ and $1 \leq j \leq q$, $w_1(l_i) = B$, $w_1(c_j) = 1$ and $w_1(r) = w_1(dc_j) = w_1(dl_i) = 0$.

Roughly, in G_1 there is a node for each clause (denoted by c_j) and for each literal (denoted by l_i) that represent the leaves of the caterpillar. The spine of the caterpillar contains dummy nodes for each clause (denoted by dc_j) and for each literal (denoted by dl_i) separated by a central node (denoted by r).

The binary tree $G_2 = (V_2, E_2)$ with weight function w_2 is defined as follows. The vertex set is $V_2 = \{r, x_i, \bar{x}_i, c_j^k, dx_i, d\bar{x}_i, dc_j^i, dc_j^{\bar{i}} \mid 1 \leq i \leq n, 1 \leq j \leq q, 1 \leq k \leq 3\}$. The edge set E_2 is given by the following equation.

$$\begin{aligned} E_2 = & \{(r, dx_n)\} \cup \\ & \{(c_j^{k'}, dc_j^k) \mid x_k, \text{ is the } k'\text{-th literal of clause } c_j\} \cup \\ & \{(c_j^{k'}, dc_j^{\bar{k}}) \mid \bar{x}_k, \text{ is the } k'\text{-th literal of clause } c_j\} \cup \\ & \{(dx_i, d\bar{x}_{i+1}) \mid 1 \leq i < n\} \cup \\ & \{(dx_i, d\bar{x}_i), (dx_i, x_{n-i+1}), (d\bar{x}_i, \bar{x}_{n-i+1}), (x_i, dc_1^i), (\bar{x}_i, dc_1^{\bar{i}}) \mid 1 \leq i \leq n\} \cup \\ & \{(dc_j^i, dc_{j+1}^i), (dc_j^{\bar{i}}, dc_{j+1}^{\bar{i}}) \mid 1 \leq i \leq n, 1 \leq j < q\} \end{aligned}$$

The weight function w_2 is defined as follows: for all $1 \leq i \leq n$, $1 \leq j \leq q$ and $1 \leq k \leq 3$, $w_2(x_i) = w_2(\bar{x}_i) = -B$ and $w_2(r) = w_2(c_j^k) = w_2(dx_i) = w_2(d\bar{x}_i) = w_2(dc_j^i) = w_2(dc_j^{\bar{i}}) = 0$

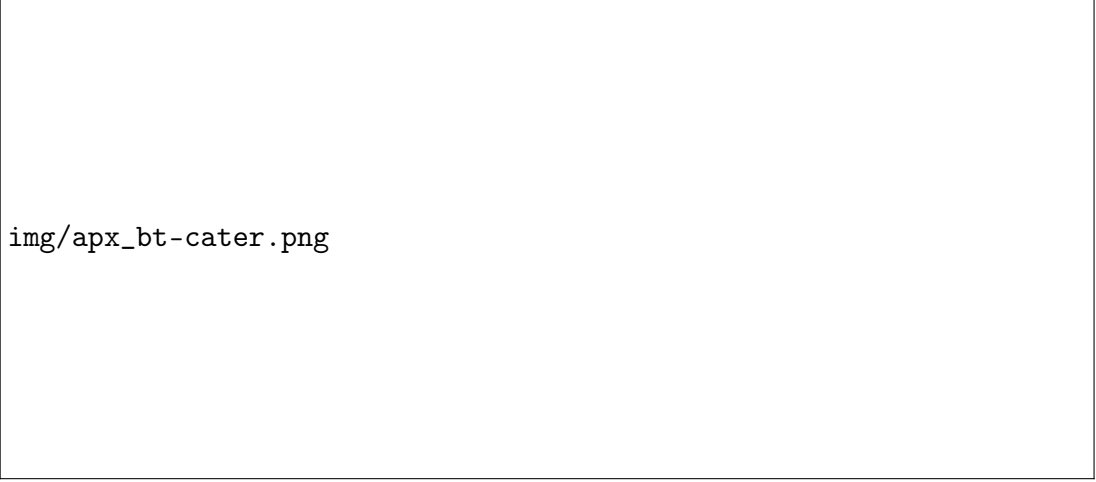
Roughly, in G_2 there is a node for each literal of each clause (denoted by c_j^k) and for each value of each literal (denoted by x_i and \bar{x}_i). Dummy nodes for literals have been duplicated (one for each value of the literal - that is dx_i and $d\bar{x}_i$). Dummy nodes for clauses have also been duplicated (one for each value of all literals - dc_j^i and $dc_j^{\bar{i}}$). The structure is not as easy to informally describe as for G_1 but the reader may refer to an illustration provided in Figure 4.1.

Finally, the mapping M is an injective function from V_1 to V_2 defined as follows.

$$\begin{aligned} M(r) &= r \\ M(l_i) &= \{x_i, \bar{x}_i\}, \text{ for all } 1 \leq i \leq n \\ M(c_j) &= \{c_j^k \mid 1 \leq k \leq 3\}, \text{ for all } 1 \leq j \leq q \\ M(dl_i) &= \{dx_i, d\bar{x}_i\}, \text{ for all } 1 \leq i \leq n \\ M(dc_j) &= \{dc_j^i, dc_j^{\bar{i}}\}, \text{ for all } 1 \leq i \leq n \text{ and } 1 \leq j \leq q \end{aligned}$$

Let us prove that this construction is indeed an L-reduction from $\text{MAX-3SAT}(B)$. More precisely, we will prove the following property.

Lemma 1. *There exists an assignment of V_n satisfying at least m clauses of C_q if and only if there exists a solution to MWCCS of weight at least m .*



img/apx_bt-cater.png

Figure 4.1: Illustration of the construction of G_1 , G_2 , and M , given $C_q = \{(x_1 \vee x_2 \vee \neg x_3), (\neg x_1 \vee x_2 \vee x_5), (\neg x_2 \vee x_3 \vee \neg x_4), (\neg x_3 \vee x_4 \vee \neg x_5)\}$. For readability, the mapping M is not drawn but represented as labels located on the nodes: any pair of nodes (one in G_1 and one in G_2) of similar inner label are mapped in M .

Proof. \Rightarrow Given an assignment \mathcal{A} of V_n satisfying m clauses of C_q , we construct a solution to MWCCS of weight m as follows.

$$\begin{aligned}
 \text{Let } V_1^* &= V_1 \setminus \{c_j \mid c_j \text{ is not satisfied by the assignment}\} \text{ and} \\
 V_2^* &= \{r\} \cup \\
 &\quad \{c_j^k \mid c_j \text{ is satisfied by its } k\text{-th literal}\} \cup \\
 &\quad \{x_i, dc_j^i \mid x_i = 1, 1 \leq j \leq q\} \cup \\
 &\quad \{\bar{x}_i, dc_j^{\bar{i}} \mid x_i = 0, 1 \leq j \leq q\} \cup \\
 &\quad \{dx_i, d\bar{x}_i \mid 1 \leq i \leq n\}.
 \end{aligned}$$

By construction, $G_1[V_1^*]$ is connected since all the vertices of the spine of the caterpillar have been kept. Moreover, $G_1[V_1^*]$ contributes $B \times n + m$ to the overall weight of the solution, that is B for each of the l_i and $+1$ for each satisfied clause. By construction, all the sub-trees rooted at x_i (resp. \bar{x}_i) are kept in $G_2[V_2^*]$ if $x_i = 1$ (resp. $x_i = 0$) in \mathcal{A} . Moreover, all the dummy nodes for literals (dx_i and $d\bar{x}_i$) and the root r have been kept. Thus, $G_2[V_2^*]$ is also connected. Furthermore, $G_2[V_2^*]$ contributes to $-B \times n$ to the overall weight of the solution since exactly one of each variable node (x_i and \bar{x}_i) has been kept. One can easily check that any node of V_1^* has a mapping counterpart in V_2^* . The overall solution is valid and of total weight m .

\Leftarrow Given any solution $\{V_1^*, V_2^*\}$ to MWCCS of weight m , we construct a solution to the MAX-3SAT(B) problem satisfying at least m clauses as follows.

First, note that we can assume that any such solution to MWCCS is *canonical*, meaning that V_2^* does not contain both vertices x_i and \bar{x}_i for all $1 \leq i \leq n$. Indeed, by contradiction, suppose there exists a solution such that $\{x_i, \bar{x}_i\} \subseteq V_2^*$ for a given $1 \leq i \leq n$. Then, $\{x_i, \bar{x}_i\}$ in G_2 induce a negative weight of $-2B$. This negative contribution can at most be compensated by the weight of the corresponding literal node in G_1 ($w_1(l_i) = B$) and at most B clause nodes in G_1 ($B \geq \sum w_1(c_j)$ where $x_i \in c_j$ or $\bar{x}_i \in c_j$) since every literal occurs in at most B clauses in C_q . Therefore, such local configuration does not provide any positive contribution to the solution and can be transformed into a better solution by removing one of the subtrees rooted in $\{x_i, \bar{x}_i\}$. We will consider hereafter that m is the weight of the resulting canonical solution. We further assume that $m > 1$ since otherwise we can build a trivial assignment $\mathcal{A} = \{c_1^1 = 1\}$ of V_n that is satisfying at least one clause of C_q .

Let \mathcal{A} be an assignment of V_n such that for all $1 \leq i \leq n$ if $x_i \in V_2^*$ then $x_i = 1$ and $x_i = 0$ otherwise. Note that, since our solution is canonical, each literal has been assigned a single boolean value in \mathcal{A} . Let us now prove that this assignment satisfies at least m clauses of C_q .

First, note that since our solution is canonical and we require any node of V_1^* to have a mapping counterpart in V_2^* , this implies that if $l_i \in V_1^*$ then its contribution (that is $w_1(l_i) = B$) is cancelled by the negative contribution of either x_i or \bar{x}_i in V_2^* (that is $w_2(x_i) = w_2(\bar{x}_i) = -B$). Therefore, the weight m of the solution can only be realized by m clause nodes of G_1 , say $\mathcal{C}_1 \subseteq V_1^*$ – since $w_1(c_j) = 1$ for all $1 \leq j \leq q$.

As already stated, to be part of the solution any node in V_1^* has a mapping counterpart in V_2^* . Thus, for each node in \mathcal{C}_1 , there should be a node of $\mathcal{C}_2 \subseteq \{c_j^k \mid 1 \leq j \leq q, 1 \leq k \leq 3\}$ in V_2^* . More precisely, by construction, any node c_j in V_1 has exactly three mapping counterparts in V_2 (that is $\{c_j^k \mid 1 \leq k \leq 3\}$) and for each $c_j \in \mathcal{C}_1$ at least one of these mapping counterparts has to belong to \mathcal{C}_2 .

Finally, since both $G_1[V_1^*]$ and $G_2[V_2^*]$ have to be connected, each node in \mathcal{C}_2 , say c_j^k , should be connected by a path to a node x_i or \bar{x}_i , say x_i , for some $1 \leq i \leq n$, in $G_2[V_2^*]$. By construction, this is the case if x_i is the k -th literal of the clause c_j for some $1 \leq k \leq 3$. Thus, \mathcal{A} is an assignment that satisfies any clause c_j such that the clause node c_j belongs to V_1^* . As already stated $|\mathcal{C}_1| = m$. \square

The above reduction linearly preserves the approximation since the weights of optimal solutions of the problems correspond and there exists an assignment of V_n satisfying at least m clauses of C_q if and only if there exists a solution to MWCCS of weight at least m . Hence, given an approximation to MWCCS, one can derive an algorithm for MAX-3SAT(B) with the same approximation ratio. Since MAX-3SAT(B), $B \geq 3$, is APX-hard Papadimitriou and Yannakakis 1991 and MAX-3SAT(B) for $B = 6$ is not approximable within factor 1.0014 Berman and Karpinski 1999, so is MWCCS, which proves Proposition 3.

Let us now prove a similar result for MWCCS problem when the mapping is a bijective function.

4.1.2 Bijective relationship function, binary tree, and graph

XXX TODO: *binary tree*, right now it's only with a general tree, need to include lots of dummies XXX

Proposition 2. *The MWCCS problem for a graph and a tree is APX-hard and not approximable within factor 1.0014 even when the mapping is a bijective function and a complete conservation (i.e. $\alpha = 1$) is required.*

Given any instance (C_q, V_n) of $\text{MAX-3SAT}(B)$, we build a graph $G_1 = (V_1, E_1)$ with weight function w_1 , a tree $G_2 = (V_2, E_2)$ with weight function w_2 and a mapping M as follows. The graph G_1 has the vertex set $V_1 = \{r, l_i, x_i, \bar{x}_i, c_j, c_j^k \mid 1 \leq i \leq n, 1 \leq j \leq q, 1 \leq k \leq 3\}$ and the edge set defined by the following equation.

$$E_1 = \{(l_i, x_i), (l_i, \bar{x}_i), (r, x_i), (r, \bar{x}_i) \mid 1 \leq i \leq n\} \cup \{(c_j, c_j^k), (r, c_j^k) \mid 1 \leq k \leq 3, 1 \leq j \leq q\}.$$

The weight function w_1 is defined as follows: for all $1 \leq k \leq 3, 1 \leq i \leq n$ and $1 \leq j \leq q$, $w_1(l_i) = B$, $w_1(c_j) = 1$ and $w_1(r) = w_1(c_j^k) = w_1(x_i) = w_1(\bar{x}_i) = 0$.

Roughly, in G_1 there is a node for each clause (denoted by c_j), for each of the three literals of each clause (denoted by c_j^k), for each literal (denoted by l_i) and for each valuation of each literal (denoted by x_i, \bar{x}_i). Clause nodes and literal nodes are separated by a central node r .

The tree G_2 is defined as follows. The vertex set is $V_2 = V_1$, the edge set is given by the following equation:

$$E_2 = \{(l_i, r), (c_j, r), (x_i, r), (\bar{x}_i, r) \mid 1 \leq i \leq n, 1 \leq j \leq q\} \cup \{(c_j^k, x_i) \mid x_i \text{ is the } k\text{-th literal of clause } c_j\} \cup \{(c_j^k, \bar{x}_i) \mid \bar{x}_i \text{ is the } k\text{-th literal of clause } c_j\}.$$

The weight function w_2 is defined as follows: for all $1 \leq k \leq 3, 1 \leq i \leq n$ and $1 \leq j \leq q$, $w_2(x_i) = w_2(\bar{x}_i) = -B$, $w_2(r) = w_2(c_j^k) = w_2(l_i) = w_2(c_j) = 0$.

Roughly, in G_2 all the nodes except the ones in $\{c_j^k \mid 1 \leq j \leq q, 1 \leq k \leq 3\}$ form a star centered in node r . The nodes representing the literal of the clause (that is c_j^k) are connected to their corresponding variable nodes (that is x_i or \bar{x}_i).

Finally, the mapping M is a bijective function from V_1 to V_2 defined as the identity (that is each node in V_1 is mapped to the node of similar label in V_2).

Let us prove that this construction is indeed an L-reduction from $\text{MAX-3SAT}(B)$. More precisely, we will prove the following property.

Lemma 2. *There exists an assignment of V_n satisfying at least m clauses of C_q if and only if there exists a solution (not necessarily optimal) to MWCCS of weight at least m .*

Proof. \Rightarrow Given an assignment \mathcal{A} of V_n satisfying m clauses of C_q , we construct a solution to MWCCS of weight m as follows.

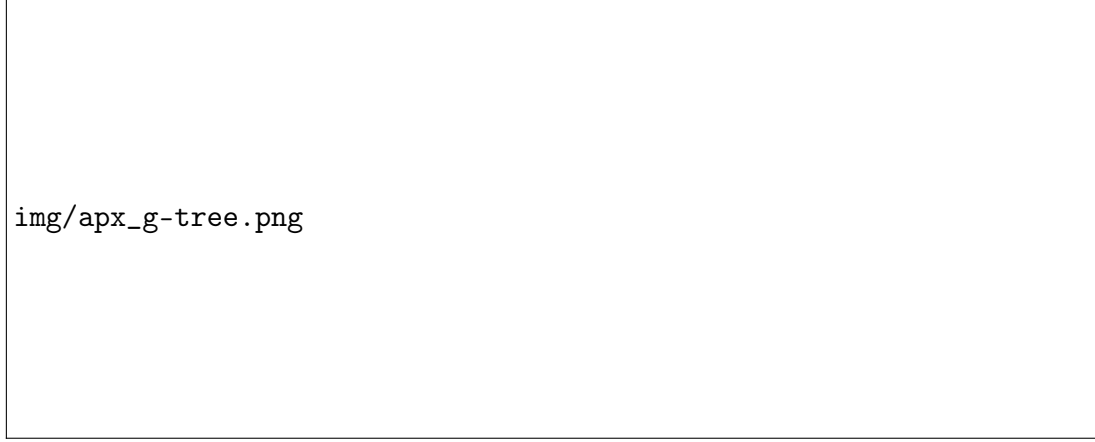


Figure 4.2: Illustration of the construction of G_1 , G_2 , and M , given $C_q = \{(x_1 \vee x_2 \vee \neg x_3), (\neg x_1 \vee x_2 \vee x_5), (\neg x_2 \vee x_3 \vee \neg x_4), (\neg x_3 \vee x_4 \vee \neg x_5)\}$. For readability, the mapping M is not drawn but deduced from the labels of the nodes; any pair of nodes (one in G_1 and one in G_2) of similar label are mapped in M .

Let $V_1^* = V_2^* = \{c_j \mid c_j \text{ is satisfied by } \mathcal{A}\} \cup \{c_j^k \mid c_j^k \text{ is satisfying } c_j \text{ by } \mathcal{A}\} \cup \{x_i \mid x_i = 1\} \cup \{\bar{x}_i \mid x_i = 0\} \cup \{r, l_i \mid 1 \leq i \leq n\}$.

By construction, $G_1[V_1^*]$ and $G_2[V_2^*]$ are connected. Moreover, $G_1[V_1^*]$ contributes $B \times n + m$ to the overall weight of the solution, that is B for each of the l_i and $+1$ for each satisfied clause, while $G_2[V_2^*]$ contributes $-B \times n$ to the overall weight of the solution since exactly one of each variable node (i.e., x_i and \bar{x}_i) has been kept. The overall solution is valid and of total weight m .

⊞ Given any solution $V^* \subseteq V_1$ to MWCCS of weight m , we construct a solution to the MAX-3SAT(B) problem satisfying at least m clauses as follows.

First, note that, as in the previous construction, we can assume that any such solution to MWCCS is *canonical* meaning that V^* does not contain both vertices x_i and \bar{x}_i for any $1 \leq i \leq n$.

Let \mathcal{A} be an assignment of V_n such that for all $1 \leq i \leq n$, if $x_i \in V^*$ then $x_i = 1$ and $x_i = 0$ otherwise. Note that, since our solution is canonical, each literal has been assigned a single boolean value in \mathcal{A} . Let us now prove that this assignment satisfies at least m clauses of C_q .

First, note that since our solution is canonical, as in the previous construction, the weight m of the solution can only be induced by m clause nodes of G_1 , say $\mathcal{C}_1 \subseteq V^*$.

Since both $G_1[V^*]$ and $G_2[V^*]$ have to be connected, any solution with $m > 1$ will include node r in V^* . Thus, for each node $c_j \in \mathcal{C}_1$ there should be a node of $\{c_j^k \mid 1 \leq k \leq 3\}$ in $G_1[V^*]$ to connect c_j to r . In $G_2[V^*]$, in order for nodes r and c_j^k to be connected, the corresponding literal node (that is x_i or \bar{x}_i), say x_i – has to be kept in V^* . By construction, this is the case if x_i is the k -th literal of clause c_j . Thus, \mathcal{A} is an assignment that satisfies any clause c_j such that the clause node c_j belongs to V^* . As already stated $|\mathcal{C}_1| = m$. □

The above reduction linearly preserves the approximation and proves Proposition 2.

4.2 Polynomial-time cases for the MWCCS problem

4.2.1 Two trees and a bijective mapping

Here we continue to explore the case with fixed $\alpha = 1$. We have proved in section 4.1.1 that on instances with two binary trees, and an injective mapping function, the problem is APX-hard. We will now consider the case of two general trees, $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$, with a bijective relationship $V_1 \cong V_2$.

The standard algorithms for MWCCS and its variants⁴ use bottom-up, usually dynamic-programming, approaches. Given an unrooted tree T , root T in one of its node to direct the edges. Then, compute recursively the maximum weight of the subtrees that unconditionally includes their root.

It would be a good characteristic of our algorithm to advance in a similar fashion, recursively solving subproblems, and progressing in both trees in locked step. However in our case, the mapping function defines a strong constraint on the possible steps since $\alpha = 1$, and a node can only be selected if its counterpart is. Furthermore, a node in one tree can have its counterpart at a completely different level in the other, leading to some difficulty in finding a viable locked-step advancement scheme. This last point make impossible to devise a local recursive procedure to find the optimal subtree. Indeed, adding a child node might require the addition of a parent by mutual requirement between the two trees and the mapping function, see section 4.2.1 for an exemple.

Figure 4.3: XXX image with mapping from root to leaf in both trees: locked-step require to take two full paths at the same time XXX

We will use the fact that we require a perfect matching to drive the steps from one of the two trees, say T_1 . Indeed, selecting a node in T_1 requires the unconditional inclusion of its counterpart in T_2 , and of all node required to connect them, this until no more nodes require to be included.

This last operation, `ConstructRequiredSet`, is defined as follows. Given two non-empty sets S_1 and S_2 of nodes already selected in V_1 and V_2 , and a node $u \in V_1 \setminus S_1$ such that u is a neighbor of a node in S_1 , $v \in V_2$ its counterpart. We will define a double recursion procedure that will work first with V_1 , then with V_2 . Add u to S_1 ; then, find the shortest path from v to any node of S_2 . Add all nodes that pertain to this shortest path to S_2 , and recursively find all shortest paths from the counterparts of those nodes to the nodes of S_1 . Continue until no more nodes need to be added.

Proposition 3. *The induced graphs $T_1[S_1]$ and $T_2[S_2]$ obtained as results of `ConstructRequiredSet` are connected.* ■


⁴Mainly cardinality-constrained, budget-constrained, and ratio-bounded, with a pseudo-polynomial algorithm for all of them exposed in the next chapter.

Proof. Indeed, since we recursively add nodes with their shortest path to already selected nodes, there exists a path connecting every pair of nodes of S_1 and of S_2 . \square

Proposition 4. *ConstructRequiredSet is a quadratic operation in the worst case.*

Proof. The shortest path in a tree is equivalent to the minimum sum subsequence problem XXX ref XXX and can be solved in linear time. We recursively add up the nodes up to $2n$. Hence the resulting complexity. \square

The main procedure is a recursive decision tree traversal that will be using the ConstructRequiredSet operation. This recursive procedure, ComputeDecisionTree is defined as follows. Given two sets S_1 and S_2 of connected nodes in V_1 and V_2 respectively. Define C as the union of all children of the nodes in S_1 . For all nodes $c \in C$, call the ConstructRequiredSet procedure with S_1 , S_2 and c , which will return S'_1 and S'_2 . Continue the construction of the decision tree by recursively calling ComputeDecisionTree with S'_1 and S'_2 until there are no more children. Note that the score of a node in the decision tree is equal to the sum of the scores of its children minus the score of the nodes that are in common.



img/IMG_20141216_163836.jpg

XXX Drawing version of the example on the board XXX

Proposition 5. *ComputeDecisionTree requires at most $O(n^3 \lg(n))$ steps.*

Proof. The decision tree contains $\prod_{i=n-1}^1 n = O(n^2)$ nodes in the worst case. To compute the score, it is required to compute the set union operation, which is an $O(n \lg(n))$ operation with tree representation of the sets. Hence the resulting number of steps. \square

Finally, note that this procedure can lead to different results depending on the starting set S_1 and S_2 . The final step is to construct the decision tree for all pairs $u, v \in V_1 \times V_2$, and select the optimal solution.

Proposition 6. *This last procedure is linear in the number of nodes.*

Proof. Indeed, since the nodes are on a one to one mapping, there exists exactly $|V_1| = |V_2|$ such pairs. \square

XXX Need to find an example and make a figure XXX

4.2.2 Polynomial-time scheme for some inputs

Here we consider the general version of the MWCCS problem where α is given as input rather than being fixed. In addition, we further relax the constraint on the relationship function, which can be any partial injective function. That is, any element of V_1 can have at most one image in V_2 (the elements of V_2 can have 0, 1, or more antecedents). Finally, we suppose that there is a polynomial number of connected induced subgraphs of G_1 .

We will consider as many candidate solution as there are connected subgraphs of G_1 , and for each one of them we will try to find the best corresponding subgraph in G_2 . The best corresponding subgraph in G_2 is the subgraph that maximizes the total weight of the candidate solution and such that at least an α -fraction of the nodes of G_1 and G_2 in the solution are M -related. For a given subgraph of G_1 , the sum of its nodes' weight is fixed, hence maximizing the total weight of the candidate solution is equivalent to finding the optimal solution in G_2 where the α -fraction constraint holds.

To solve this problem, we introduce a new problem, the RATIO-BOUNDED MAXIMUM-WEIGHT CONNECTED SUBGRAPH (RB-MWCS) problem, for which the formal definition and detailed analysis are provided in the next chapter. Informally, it consists in a variant of the MWCS problem where an additional contribution function is associated to each node and where an additional ratio constraint is introduced⁵.

The reduction to this new subproblem is as follows. The contribution function needs to *encode* the relationship function between the nodes of both G_1 and G_2 . To do so, and since we fixed a partial solution to the problem in the form of the subgraph of G_1 , we note for each node of G_2 its number of inverse images in G_1' plus one if and only if at least one exists, zero otherwise. The reason we need to make the distinction between the two cases is that when there is no counterpart in G_1 , selecting a node in the optimal solution in G_2 will not increase the number of mapped node overall. However, selecting a node in G_2 for which there exists at least one counterpart will increase the number of mapped node by the number of counterpart, plus one for the selected node itself.

⁵not unlike the *budget-constrained* variant of the MWCS problem, cf. next chapter for more details

Formally, given a connected subgraph $G'_1 = (V'_1, E'_1)$ of G_1 , we define the corresponding G_2 antecedent function $a: V_2 \rightarrow \mathbb{N}$ to be $a(v) = |\{v, u \mid M(u, v), u \in V'_1\}|$. The contribution function $c: V_2 \rightarrow \mathbb{N}$ is defined as follows:

$$c(v) = \begin{cases} a(v) + 1 & \text{if } a(v) > 0, \\ 0 & \text{otherwise.} \end{cases}$$

Given $G_2 = (V_2, E_2)$, its weight-function w_2 and its contribution function c , the problem now corresponds to the discovery of the connected subgraph of maximum weight such that:

$$\begin{aligned} \sum_{v \in V_2^*} c(v) &\geq \alpha \times (|V'_1| + |V_2^*|) \\ \sum_{v \in V_2^*} c(v) - \alpha \times |V'_1| &\geq \alpha \times |V_2^*| \end{aligned}$$

Where $\alpha \times |V'_1|$ is constant.

Finally, given the optimal score of all candidate solutions, for which there are one for each subgraphs G'_1 , the optimal solution to the MWCCS problem is the one candidate solution among them which has the best score.

Clearly, constructing a contribution function for G_2 given a subgraph G'_1 is a linear time operation. Choosing the best candidate solution is as time consuming as enumerating all subgraphs of G_1 , which is a polynomial time operation in this context. The complexity of this algorithm hence depends on the difficulty to solve the RB-MWCS subproblem.

Chapter 5 provides a detailed analysis of this problem with a more general contribution function. It provides pseudo-polynomial scheme for all classes of input graphs up to bounded treewidth. However, when the contribution function is enumerable with a polynomial upper bound, and such is the case by construction here, the scheme actually becomes a polynomial-time algorithm. Thus, there exists a polynomial-time algorithm to solve MWCCS when one of the graph is polynomially enumerable, the second is bounded treewidth, and the relationship is a partial injective function, for any $\alpha \in [0, 1]$.

Chapter 5

The Ratio-Bounded MWCS

In this chapter, XXX explain XXX.

A slightly more general version¹ of the problem introduced at the end of the previous chapter is called the RATIO-BOUNDED MAXIMUM-WEIGHT CONNECTED SUBGRAPH (RB-MWCS) problem, and in the general case is defined formally as follows.

RB-MWCS: Given a node-weighted graph $G = (V, E)$, its node-weighting function $w: V \rightarrow \mathbb{R}$, its contribution function $c: V \rightarrow \mathbb{R}$, a ratio $\alpha \in [0, 1]$ and a constant $C \in \mathbb{R}$, find a subset $V^* \subseteq V$ such that:

1. the induced graph $G[V^*]$ is connected, and
2. the ratio of the sum of contributions plus some constant over the number of nodes in the solution is greater than or equal to α , that is:
$$\sum_{v \in V^*} c(v) + C \geq \alpha \times |V^*|$$
, and
3. $\sum_{v \in V^*} w(v)$ is maximum.

Proposition 7. *RB-MWCS is as difficult as MWCS.*

Proof. Indeed, when $\forall v \in V, c(v) = 1$, the ratio $\sum_{v \in V^*} c(v)/|V^*| = 1 \geq \alpha$, and the MWCS and RB-MWCS problems are equivalent. \square

5.1 A more general variant of the Budget-Constrained MWCS

The BUDGET-CONSTRAINED MAXIMUM-WEIGHT CONNECTED SUBGRAPH, also named the BUDGETED MWCS (B-MWCS) is formally defined as follow.

B-MWCS: Given a node-weighted graph $G = (V, E)$, its node-weighting function $w: V \rightarrow \mathbb{R}$, its cost function $\text{cost}: V \rightarrow \mathbb{R}^+$, and a budget $B \in \mathbb{R}^+$, find the subset $V^* \subseteq V$ such that:

1. the induced graph $G[V^*]$ is connected, and

¹the contribution function is more expressive here

2. the sum of the costs does not exceed the allocated budget B , that is:

$$\sum_{v \in V^*} \text{cost}(v) \leq B, \text{ and}$$

3. $\sum_{v \in V^*} w(v)$ is maximum.

Note that appart from the cost and ratio constraints, the two problems are stricly equivalent.

Proposition 8. *The ratio constraint $\sum_{v \in V^*} c(v) + C \geq \alpha \times |V^*|$ of the RB-MWCS problem is a generalization of the costs constraint $\sum_{v \in V^*} \text{cost}(v) \leq B$ of the B-MWCS problem.*

Proof.

$$\begin{aligned} \sum_{v \in V^*} c(v) + C &\geq \alpha \times |V^*| \\ \sum_{v \in V^*} c(v) + C &\geq \sum_{v \in V^*} \alpha \\ \sum_{v \in V^*} c(v) - \sum_{v \in V^*} \alpha &\geq -C \\ \sum_{v \in V^*} c(v) - \alpha &\geq -C \\ \sum_{v \in V^*} \alpha - c(v) &\leq C \\ \sum_{v \in V^*} c'(v) &\leq C \end{aligned}$$

The range of the function c' is a superset of the range of the function cost since the later is positive. The same argument is made for B-MWCS's constant B which is positive where the constant C is real. \square

Proposition 9. *There exists a linear reduction from B-MWCS to RB-MWCS problem, hence the latter is more general.* \square

Proof. Given an B-MWCS problem,

1. the graph $G = (V, E)$ do not change, neither does the weight function w ,
2. define the cost function $c(v) = \alpha - \text{cost}(v)$, $\forall \alpha \in [0, 1]$, and
3. the constant $C = B$.

This defines a suitable reduction from B-MWCS to RB-MWCS. \square

5.2 Pseudo-polynomial algorithms from MWCS

XXX Argument about the reasoning with $c: V \rightarrow \mathbb{Z}$ instead of $c: V \rightarrow \mathbb{R}$, which 1] is valid for our reduction in chap 4 and 2] holds for B-MWCS XXX

5.2.1 Explicit algorithms for trees and cactii

5.2.2 For bounded treewidth graphs

Conclusion

Bibliography

- Álvarez-Miranda, Eduardo, Ivana Ljubić, and Petra Mutzel (2013a). “The maximum weight connected subgraph problem”. In: *Facets of Combinatorial Optimization*. Springer, pp. 245–270 (cit. on pp. 9, 11, 15).
- (2013b). “The rooted maximum node-weight connected subgraph problem”. In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. Springer, pp. 300–315 (cit. on p. 11).
- Backes, Christina, Alexander Rurainski, Gunnar W Klau, Oliver Müller, Daniel Stöckel, Andreas Gerasch, Jan Küntzer, Daniela Maisel, Nicole Ludwig, Matthias Hein, et al. (2012). “An integer linear programming approach for finding deregulated subgraphs in regulatory networks”. In: *Nucleic acids research* 40.6, e43–e43 (cit. on p. 10).
- Berman, Piotr and Marek Karpinski (1999). *On some tighter inapproximability results*. Springer (cit. on p. 21).
- Boykov, Yuri and Vladimir Kolmogorov (2004). “An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 26.9, pp. 1124–1137 (cit. on p. 16).
- Camerini, PM, G Galbiati, and F Maffioli (1979). “On the complexity of Steiner-like problems”. In: *Proceedings 17th Ann. Allerton Conf. on Communication, Control, and Computing*. Department of Electrical Engineering and the Coordinated Science Laboratory, University of Illinois Urbana, Ill, pp. 969–977 (cit. on p. 9).
- Chang, Tse-Wen (1983). “Binding of cells to matrixes of distinct antibodies coated on solid surface”. In: *Journal of immunological methods* 65.1, pp. 217–223 (cit. on p. 7).
- Chang, T.W. (1986). *Matrix of antibody-coated spots for determination of antigens*. US Patent 4,591,570. URL: <http://www.google.com/patents/US4591570> (cit. on p. 7).
- (1989). *Immunoassay device enclosing matrixes of antibody spots for cell determinations*. US Patent 4,829,010. URL: <https://www.google.com/patents/US4829010> (cit. on p. 7).
- (1992). *Antibody matrix device and method for evaluating immune status*. US Patent 5,100,777. URL: <https://www.google.com/patents/US5100777> (cit. on p. 7).
- Conrad, Jon, Carla P Gomes, Willem-Jan van Hoeve, Ashish Sabharwal, and Jordan Suter (2007). “Connections in networks: Hardness of feasibility versus optimality”. In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. Springer, pp. 16–28 (cit. on pp. 9, 10).

- Dilkina, Bistra and Carla P Gomes (2010). “Solving connected subgraph problems in wildlife conservation”. In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. Springer, pp. 102–116 (cit. on pp. 10, 15).
- Dittrich, Marcus T, Gunnar W Klau, Andreas Rosenwald, Thomas Dandekar, and Tobias Müller (2008). “Identifying functional modules in protein–protein interaction networks: an integrated exact approach”. In: *Bioinformatics* 24.13, pp. i223–i231 (cit. on pp. 10, 11).
- Feigenbaum, Joan, Christos Papadimitriou, and Scott Shenker (2001). “Sharing the cost of multicast transmissions”. In: *Journal of Computer and System Sciences* 63.1, pp. 21–41 (cit. on p. 9).
- Garey, Michael R and David S Johnson (1979). “Computers and intractability: a guide to the theory of NP-completeness”. In: *San Francisco, LA: Freeman* (cit. on p. 9).
- Golub, Todd R, Donna K Slonim, Pablo Tamayo, Christine Huard, Michelle Gaasenbeek, Jill P Mesirov, Hilary Collier, Mignon L Loh, James R Downing, Mark A Caligiuri, et al. (1999). “Molecular classification of cancer: class discovery and class prediction by gene expression monitoring”. In: *Science* 286.5439, pp. 531–537 (cit. on p. 8).
- Gomes, Carla P, Willem-Jan Van Hoeve, and Ashish Sabharwal (2008). “Connections in networks: A hybrid approach”. In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. Springer, pp. 303–307 (cit. on p. 10).
- Hauptmann, Mathias and Marek Karpinski (2014). *A compendium on steiner tree problems* (cit. on p. 8).
- Hochbaum, Dorit S and Anu Pathria (1994). “Node-optimal connected k-subgraphs”. In: *manuscript, UC Berkeley* (cit. on p. 10).
- Hume, Thomas, Hayssam Soueidan, Macha Nikolski, and Guillaume Blin (2015). “Approximation Hardness of the Cross-Species Conserved Active Modules Detection Problem”. In: *SOFSEM 2015: Theory and Practice of Computer Science*. Springer, pp. 242–253 (cit. on p. 17).
- Ideker, Trey, Owen Ozier, Benno Schwikowski, and Andrew F Siegel (2002). “Discovering regulatory and signalling circuits in molecular interaction networks”. In: *Bioinformatics* 18.suppl 1, S233–S240 (cit. on pp. 8, 9, 11).
- Johnson, David S (1985). “The NP-completeness column: an ongoing guide”. In: *Journal of Algorithms* 6.1, pp. 145–159 (cit. on pp. 9, 14).
- Karp, Richard M (1972). *Reducibility among combinatorial problems*. Springer (cit. on pp. 8–10).
- Lee, Heungsoon Felix and Daniel R Dooly (1998). “Decomposition algorithms for the maximum-weight connected graph problem”. In: *Naval Research Logistics* 45.8, pp. 817–837 (cit. on p. 10).
- Magnanti, Thomas L and Laurence A Wolsey (1995). “Optimal trees”. In: *Handbooks in operations research and management science* 7, pp. 503–615 (cit. on p. 15).
- Mitra, Koyel, Anne-Ruxandra Carvunis, Sanath Kumar Ramesh, and Trey Ideker (2013). “Integrative approaches for finding modular structure in biological networks”. In: *Nature Reviews Genetics* 14.10, pp. 719–732 (cit. on pp. 8, 11).

- Noort, Vera van, Berend Snel, and Martijn A Huynen (2003). “Predicting gene function by conserved co-expression”. In: *Trends in Genetics* 19.5, pp. 238–242 (cit. on p. 8).
- Papadimitriou, Christos H and Mihalis Yannakakis (1991). “Optimization, approximation, and complexity classes”. In: *Journal of Computer and System Sciences* 43.3, pp. 425–440 (cit. on pp. 5, 21).
- Schena, Mark, Dari Shalon, Ronald W Davis, and Patrick O Brown (1995). “Quantitative monitoring of gene expression patterns with a complementary DNA microarray”. In: *Science* 270.5235, pp. 467–470 (cit. on p. 7).
- Southern, Edwin Mellor (1975). “Detection of specific sequences among DNA fragments separated by gel electrophoresis”. In: *Journal of molecular biology* 98.3, pp. 503–517 (cit. on p. 7).
- Szklarczyk, Damian, Andrea Franceschini, Stefan Wyder, Kristoffer Forslund, Davide Heller, Jaime Huerta-Cepas, Milan Simonovic, Alexander Roth, Alberto Santos, Kalliopi P Tsafou, et al. (2014). “STRING v10: protein–protein interaction networks, integrated over the tree of life”. In: *Nucleic acids research*, gku1003 (cit. on p. 8).
- Vergis, A (1983). “manuscript” (cit. on p. 9).
- Yamamoto, Takanori, Hideo Bannai, Masao Nagasaki, and Satoru Miyano (2009). “Better decomposition heuristics for the maximum-weight connected graph problem using betweenness centrality”. In: *Discovery Science*. Springer, pp. 465–472 (cit. on p. 11).