Scientific Annals of Computer Science vol. 32 (1), 2022, pp. 109–136
doi: 10.7561/SACS.2022.1.109

Fixed Parameter Algorithms and Hardness of Approximation Results for the Structural Target Controllability Problem[1]

Eugen Czeizler[2]
Alexandru Popa[3]
Victor Popescu[4]

Abstract

Recent research has revealed new and unexpected applications of network control science within biomedicine, pharmacology, and medical therapeutics. These new insights and new applications generated in turn a rediscovery of some old, unresolved algorithmic problems. One of these problems is the Structural Target Control optimization problem, known in previous literature also as Structural Output Controllability problem, which is defined as follows. Given a directed network and a target subset of nodes, the task is to select a small (or the smallest) set of nodes from which the target can be independently controlled, i.e., there exists a set of paths from the selected set of nodes (called driver nodes) to the target nodes such that no two paths intersect at the same distance from their targets. Recently, Structural Target Control optimization problem has been shown to be NP-hard, and several heuristic algorithms were introduced and analyzed, both on randomly generated networks, and on biomedical ones.

[1]A preliminary version of this paper was presented at the 5th International Conference on Algorithms for Computational Biology (AlCoB 2018)

[2]Åbo Akademi University, Turku, Finland, Email: eugen.czeizler@abo.fi
[3]University of Bucharest, Romania, Email: alexandru.popa@fmi.unibuc.ro
[4]Åbo Akademi University, Turku, Finland, Email: vpopescu@abo.fi

In this paper, we show that the Structural Target Controllability problem is fixed parameter tractable when parameterized by the number of target nodes. We also prove that the problem is hard to approximate at a factor better than $O(\log n)$. Taking into consideration the real case formulations of this problem we identify two more parameters which are naturally constrained by smaller bounds: the maximal length of a controlling path and the size of the set of nodes from which the control can start. With these new parameters we provide an approximation algorithm which is of exponential complexity in the size of the set of nodes from which the control can start and polynomial in all the other parameters.

Keywords: Structural control, Network control, Optimization algorithm, Fixed parameter algorithm, NP-hardness, Linear networks.

1 Introduction

The network control research field has been investigated for more than 50 years, with some of its algorithmic questions only recently being able to be solved. The general topic is concerned with the optimization of output intervention needed in order to drive a linear, time-invariant, dynamical system from an arbitrary initial state, to a precise final configuration, in finite time. Although many real-life dynamical systems tend not to be linear, most of these systems are known to be well approximated by such dynamics, or could behave as such in specific conditions, such as at their steady state. Inquiries into this field have been initiated in the 60's and 70's, see, e.g. [16, 13, 24]. However, only in 2011 Liu et al. [17] proved that the full network control optimization problem can be solved in polynomial time via a reduction to the maximum matching problem in directed graphs. The result received a lot of interest, and sparked a renewal of the field. Since then, the network control theory and its newly discovered results have been successively applied to the study of control over power grid networks [12], of biomedical signaling processes [14, 11, 27], and even the control of social networks [15, 17].

Driven by this new insight into the field as well as by its new applications into the current world of Big (or just Large) Data, researchers have realized that full control can sometimes still be too expensive. For example, network control theory has been recently applied in the case of cancer-related biomedical networks [14, 11], with the aim of using known drugs in order to drive the system towards a more favorable state. Thus, researchers aimed to use the protein signaling network in order to drive cancerous

cells towards apoptosis, i.e., programmed cell death. However, the full controllability of sparse homogeneous networks, such as many bio-medical networks (e.g., gene signaling networks, metabolic networks, gene regulating networks, etc.) requires a lot of effort, sometimes needing a direct outside control over up to 70% of the initial nodes of the network [14, 17]. As in these cases an outside control equivalents to the use of specific drugs, and since these protein networks contain up to 2-3 thousands nodes, a 70% direct outside control would imply an unfeasible solution. Thus, we have a new controllability problem, that is a variant of the initial control-theory problem, namely that of target-control. Instead of enforcing the control of the entire network, one alternative goal is to optimize the outside intervention needed to control only a well-specified target, i.e., a subset of the initial network. The aforementioned goal proves to be particularly well-fitted with the study of protein signaling networks, as recent research has emphasized the existence of disease-specific essential genes, i.e., disease-specific sets of genes/proteins which, if knocked down, would drive the corresponding cells to apoptosis [2, 28, 31]. As is the case, new formulations lead to new problems. The Structural Target Control (optimization) problem (STC) [10, 4] asks to provide an optimum amount of outside intervention in order to drive a linear dynamical system from any initial state to a desired final state of the chosen targets.

Contrary to the full network control case, the Structural Target Controllability problem was proved to be NP-hard [4]. Several heuristic approaches have been implemented and applied to the study of biomedical networks [10, 4, 14, 11]. However, approximation algorithms for this problem are not known.

Assuming the widely believed conjecture, that $P \neq NP$, no polynomial time exact algorithms exist for any NP-hard problems. Thus, there are several alternative methods to tackle the difficulty of these problems, such as *approximation algorithms* and *fixed parameter algorithms*. Approximation algorithms run in polynomial time and provide a suboptimal solution. Nevertheless, unlike heuristic algorithms, approximation algorithms guarantee that on every input instance the solution they return is within a certain factor of the optimal solution. For example, a 2-approximation algorithm for a minimization problem guarantees that on every input the solution returned is at most twice the cost of the optimal solution on that input. However, some problems, such as the one studied in this paper, might not have approximation algorithms with a constant approximation factor, unless $P = NP$. See [26] for a textbook on approximation algorithms.

In practice, many problems have parameters that are typically much smaller than the input size. We can exploit the existence of these parameters in order to design faster algorithms for these problems. *Parameterized complexity* [9, 5] aims to classify problems according to various parameters that are independent of the size of the input. A fixed parameter algorithm runs in time $f(k)O(n^c)$, where n is the input size, c is a constant, and k is the value of a parameter (independent of the input size). A problem is termed fixed parameter tractable (FPT) if it has an FPT algorithm.

In this paper we show that the Structural Target Controllability problem is fixed parameter tractable when parameterized by the number of target nodes. Also, if a second parameter is allowed, namely the maximal length of a controlling path (which is known in practice to have low values), the resulting fixed parameter algorithm has a considerably improved complexity. Moreover, we formally prove that the STC problem is hard to approximate within a factor better than $O(\log n)$.

Taking into consideration the medical and pharmaceutical insights on how this problem is formulated in the biomedical setting, we identify yet another parameter which is bounded by a lower value. This parameter is the size of the set of nodes from the network which can potentially be influenced by outside interventions, i.e., from which we can select our controlling nodes. These nodes correspond to known proteins which are targets of actual drugs, aka. drug-targets. The resulting formulation of the problem, i.e., the Driver Restricted STC has itself a fixed parameter algorithm, which has a considerably improved time complexity. However, even with the above mentioned additional constraint, the problem is intractable for real-case networks consisting of 100+ nodes. Thus, we design an approximation algorithm, which is of exponential complexity only in the size of the set of nodes from which the control can start and low polynomial in all the other parameters of the problem, i.e., the total number of nodes of the network, the size of the target set, and the maximal length of a controlling path.

2 Notation and Preliminaries

A *linear, time invariant dynamical system* (LTIS) is a system

$$\frac{dx(t)}{dt} = Ax(t) \tag{1}$$

where $x(t) = (x_1(t), \ldots, x_n(t))^T$ is the n-dimensional vector describing the system's state at time t, and $A \in \mathbb{R}^{n \times n}$ is the time-invariant *state transition*

matrix. The elements in x are called the *variables* of the system. We denote by X the set of these variables.

The external control over the system is performed through the action of m external *driver nodes*, $u(t) = (u_1(t), \ldots, u_m(t))^T$. Their influence over the n variables of the system is described by the time-invariant *input matrix* $B \in R^{n \times m}$; then the LTIS (1), now denoted as (A, B), becomes:

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t) \tag{2}$$

Let $T \subseteq X$, $T = \{t_1, \ldots, t_k\}$ for some $k \leq n$ be a subset of a particular interest for the variables X, a.k.a., *the target set*. We say that the LTIS (A, B) is *T-target controllable* if for any initial state of the variables in X and any target variables, there exists a time-dependent input vector $u(t) = (u_1(t), \ldots, u_m(t))^T$ that can drive the system in finite time from its initial state to a state in which the target variables are in the desired final setup. We associate to the k-target set T the characteristic matrix $C_T \in \{0, 1\}^{k \times n}$ where $C_T(i, j) = 1$ iff $i = j$ and $i, j \in T$ (otherwise, $C_T(i, j) = 0$). It is known, see e.g. [10], that a system (A, B) is T-target controllable if and only if

$$\operatorname{rank} \mathcal{OC}(A, B, C_T) = |T| \tag{3}$$

where the matrix $\mathcal{OC}(A, B, C_T) := [C_T B \mid C_T AB \mid C_T A^2 B \mid \ldots \mid C_T A^{n-1} B]$ is called the *controllability matrix*.

In the particular case when the target is the entire n variable set X, the above condition translates to the well known Kalman's condition for full controllability [13], i.e., an LTIS (A, B) is (fully) controllable if and only if $\operatorname{rank}[B \mid AB \mid A^2 B \mid \ldots \mid A^{n-1} B] = n$.

The notion of target controllability and the focus of imposing a controlling effect only on a subset of the variables of the system, has been introduced and studied only recently, see e.g., [10, 4, 14, 11]. However, this notion can be seen as a special case of output controllability, a topic which received considerate attention in the 80's and 90's, see. e.g. the works of Poljak and Murota [21, 22, 20].

Although the control methodology seems to be very dependent on the input data, i.e., the transition matrix A, it turns out that this is not the case. We say that an LTIS (A, B) is *T-structurally target controllable* (with respect to a given size-k target set T) if there exists a time-dependent input vector $u(t) = (u_1(t), \ldots, u_m(t))^T$ and matrices A and B with non-zero values, that can drive the state of the target nodes to any desired output in finite time. A

deep result of [16, 24] shows that a system is structurally target controllable if and only if it is target controllable for all structurally equivalent matrices A and B , except a so-called "thin" set of matrices; we say that two matrices are *structurally equivalent* iff they have the same dimensions and differ only on their non-zero values.[5] Thus, almost all matrices A and B are "a good choice". According to equation (3) above, for a k-sized target T, a system (A, B) is structurally T-target controllable if and only if there exist values for the non-zero entries in A, B such that $\operatorname{rank} \mathcal{OC}(A, B, C_T) = |T| = k$.

It was shown in [4] that from a practical perspective, it is more meaningful to analyze the controllability optimization problem from the point of view of minimizing the number of driven nodes. Thus, we focus on this particular formulation of the optimization problem. Thus, we impose that each driver node is connected to exactly one driven node, i.e., in the matrix representation of the above network we require that the input matrix B contains exactly one non-zero element on each column. We define the notion of optimization for structural target controllability in case of LTIS as follows:

Definition 1 (The Structural Target Control (Optimization) problem in case of LTIS)

Input: *The size-n variable set X, the associate transition matrix A of size $n \times n$, and a size-k target subset $T \subseteq X$, with $k \leq n$.*

Output: *Matrix B of size $n \times m$ such that*

- *(a) every column of B contains exactly one non-zero value,*
- *(b) $Srank\mathcal{OC}(A, B, C_T) = k$, where $Srank\mathcal{OC}(A, B, C_T)$, is the generic rank (or structural rank) of the structural matrix $\mathcal{OC}(A, B, C_T)$, i.e., the maximal value for the rank of $\mathcal{OC}(A, B, C_T)$ for matrices $\overline{A}$, $\overline{B}$, and $\overline{C_T}$ that have non-zero values on the non-empty entries of A, B, and C_T, respectively.*
- *(c) m (i.e., the number of columns of B) is minimum among all feasible matrices.*

It is known, see e.g. [21, 22], that the structural controllability problem has a counterpart formulation in terms of graphs/networks. Given an LTIS

[5] It is beyond the goal of this paper to define the topological notion of thin sets; we only give here the intuition that such sets consist of isolated cases that may be easily replaced with nearby favorable cases.

(A, B), we associate to it the graph $G_{(A,B)} = (V, E)$ where the n variables of the system $\{x_1, \ldots, x_n\}$ and the size-m external controller $\{u_1, \ldots, u_m\}$ are the nodes of the graphs, while directed edges correspond to the non-zero values in the state transition matrix and input matrix, respectively. That is, there exists a directed edge from the node corresponding to variable x_i to the node corresponding to x_j if and only if $A(x_j, x_i) \neq 0$.[6] Similarly, there exists a directed edge from u_i to x_j if and only if $B(x_j, u_i) \neq 0$. The nodes $\{u_1, \ldots, u_m\}$ are called *driver nodes*, while the nodes x_j such that there exists i with $B(x_j, u_i) \neq 0$ are called the *driven nodes* of the network. In the literature, the driver and the driven nodes are sometimes known as *input* and *controlled* nodes [10, 17]. To a rough understanding, the difference between driver and driven nodes is as follows. The set of driver nodes is describing the complexity of an outside controller, assuming this controller can interact/influence independently several well specified nodes of the network. Meanwhile, the set of driven nodes provides the exact collection of network nodes that are used in order to ultimately control the entire set of targets. From an algebraic perspective, the number of driver nodes is given by the number of (nonzero) columns of the control matrix B, while the number of driven nodes is given by the number of nonzero rows of B.

Given an LTIS (A, B) and its associated graph $G_{(A,B)} = (V, E)$, the n variables of the system are (all) structurally controllable from the m-sized input controller u (and control matrix B) if and only if we can select a set of n directed paths from driver nodes as starting points (we denote this set as $\mathcal{U}$) to each of the network nodes, as ending points, such that no two paths would intersect at the same distance d from their end points. The above formulation is closely related to the concepts of *linking* and *dynamic graph* as investigated in [22, 21]. In case of the target controllability problem, for a given target set $T = \{t_1, t_2, \ldots, t_k\} \subseteq X$, the above graph formulation is naturally adjusted as follows. We introduce k new *output nodes* $\mathcal{C}_\mathcal{T} = \{c_1, c_2, \ldots, c_k\}$ (also denoted as $\mathcal{C}$ when clear from the context) and edges (t_i, c_i), for all $1 \leq i \leq k$. Note that the output matrix C_T describes exactly the above mapping. Now, the objective is to find a path family containing k directed paths, connecting all the driver nodes (as start-points) to the output nodes (as end-points), such that no two paths intersect at the same distance d from their end-points. In contrast to the case of full control, the graph condition is only necessary for target control, but not sufficient [21].

[6] We implicitly interchange the usage of x_i and i for matrix indices.

However, as investigated in [4], only in very restrictive cases the existence of such a path family does not translate into the algebraic definition of structural control. Thus, from all practical purposes, the algorithmic process of finding such a family of k directed paths is equivalent to verifying that the system is structural target controllable.

We give now the formal definition of the Structural Target Control (Optimization) problem in terms of graph theory.

Definition 2 (The Structural Target Control (Optimization) problem in terms of graphs (in short STC))

The input consists of a directed graph $G = (V, E)$ and a set of nodes $T = \{t_1, t_2, \ldots, t_k\} \subseteq V$. The goal is to find a set of nodes $S \subseteq V$ of minimum cardinality that controls T. A set $S \subseteq V$ controls T if there exists k paths, $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_k$, where $\mathcal{P}_i$ starts with a node in S and ends with t_i and any two paths $\mathcal{P}_i$ and $\mathcal{P}_j$ do not intersect at the same distance d from their endpoints.

3 Fixed Parameter Algorithms

In this section we prove that the STC problem is fixed parameter tractable, when parameterized by several variables of our problem. First, we show that one parameter, namely the number of target nodes $|T| = k$, suffices in generating such a fixed parameter algorithm. On the other hand, considering the practical instances that motivate this problem, namely the targeted control of human protein signaling networks in cancer, we identify several other variables of this problem which are known to have significantly lower values, i.e., one or even two orders of magnitude lower than the total number of input nodes. Thus, we design more efficient FPT algorithms for the structural target control optimization problem using several other parameters.

3.1 A One-parameter STC Algorithm

In this subsection we present the FPT algorithm parameterized only by $|T| = k$, the size of the target set. Our algorithm uses as a subroutine an algorithm for the Set Cover problem and, thus, we first define the Set Cover problem.

Definition 3 (Set Cover) *Given a universe of elements* $\mathcal{U} = \{u_1, \ldots, u_k\}$ *and a family consisting of* n *subsets of* $\mathcal{U}$, $\mathcal{S} = \{S_1, \ldots, S_n\}$, *find the smallest sub-collection* $\mathcal{S}' \subseteq \mathcal{S}$, *such that the union of all the sets* $\mathcal{S}'$ *is* $\mathcal{U}$.

Informally, our algorithm carries out the following steps. Firstly, for each node v in the input graph, we compute all possible subsets of T that v can control. Since $|T| = k$, there can be at most 2^k such subsets for each node v. Then, we enumerate over all possible subsets of 2^T (notice that there are precisely 2^{2^k} such subsets). For each such subset $\mathcal{D} \subseteq 2^T$, we check if there exists a collection of $|\mathcal{D}|$ nodes, such that each node controls precisely one set in $\mathcal{D}$. If so, we solve exactly the set cover instance $(\mathcal{D}, T)$ and store the solution if it is better than the previously found solutions (i.e., needs less nodes than the previous solutions to control the target nodes). Algorithm 1 describes our procedure in detail.

Algorithm 1 An FPT algorithm for the STC problem

Input: An directed graph $G = (V, E)$ and a set of nodes $T \subseteq V$, $|T| = k$
Output: A set of nodes $S \subseteq V$ of minimum cardinality that controls T.

1. For every node $v \in V$, compute all possible sets of target nodes $C_v \in 2^T$ that v can control at the same time.
2. $OPT := \infty$, $S = \emptyset$
3. For every $\mathcal{D} = \{D_1, D_2, \ldots, D_\ell\} \subseteq 2^T$ such that there exist nodes $v_1, v_2, \ldots, v_\ell$ such that $C_{v_1} = D_1, C_{v_2} = D_2, \ldots, C_{v_\ell} = D_\ell$, do:
   (a) Solve exactly the set cover problem on instance $(\mathcal{D}, T)$.
   (b) Let $\mathcal{D}' = \{D_{u_1}, D_{u_2}, \ldots, D_{u_x}\}$ be the sets in the optimal set cover. If $x < OPT$, then $OPT := x$ and $S := \{u_1, u_2, \ldots, u_x\}$.

return S

Before we show the correctness of Algorithm 1, we prove the following lemma. Informally, Lemma 1 allows us to perform step 3b) of Algorithm 1, that is to safely combine the sets controlled by two or more different nodes.

Lemma 1 *Assume that the sets* $D_{u_1}, D_{u_2}, \ldots, D_{u_x} \subseteq V$ *are controlled by the nodes* $u_1, u_2, \ldots, u_x \in V$, *respectively. Then, the set* $\mathcal{S} := \{u_1, u_2, \ldots, u_x\}$ *controls* $D_{u_1} \cup D_{u_2} \cup \cdots \cup D_{u_x}$.

Proof: Observe that it is enough to prove the lemma for two subsets D_{u_1} (simply denoted as D_1 in the following) and D_{u_2} (denoted as D_2) controlled by the nodes u_1 and u_2, respectively; the generalization follows immediately.

Let A be the generic matrix associated to our graph, and B_1, B_2 be the column vectors describing the action of input nodes u_1 and u_2 over the network. Then, by Equation (3) above and Definition 1, there exist values for the non-zero entries of A, B_1 and B_2, such that $\operatorname{rank}\mathcal{OC}(A, B_1, C_{D_1}) = \operatorname{rank}[C_{D_1}B_1|C_{D_1}AB_1|\dots|C_{D_1}A^{n-1}B_1] = |D_1|$ and $\operatorname{rank}\mathcal{OC}(A, B_1, C_{D_2}) = \operatorname{rank}[C_{D_2}B_2|C_{D_2}AB_2|\dots|C_{D_2}A^{n-1}B_2] = |D_2|$.

Let $M_1, M_2, \dots, M_{|D_1|}$ and $N_1, N_2, \dots, N_{|D_2|}$ denote some linear independent columns from $\mathcal{OC}(A, B_1, C_{D_1})$ and $\mathcal{OC}(A, B_2, C_{D_2})$, respectively, such that $\det(M_1|M_2|\dots|M_{D_1}) \neq 0$ and $\det(N_1|N_2|\dots|N_{D_2}) \neq 0$.

Let $D{=}D_1{\cup}D_2$, $B{=}[B_1|B_2]$, and let investigate the rank of $\mathcal{OC}(A, B, C_D)$: $|D| \geq \operatorname{rank}\mathcal{OC}(A, B, C_D) = \operatorname{rank}[C_DB|C_DAB|C_DA^2B|\dots|C_DA^{n-1}B] = \operatorname{rank}[C_DB_1|C_DAB_1|\dots|C_DA^{n-1}B_1|\dots|C_DB_2|C_DAB_2|\dots\ C_DA^{n-1}B_2|] \geq \operatorname{rank}[\overline{M_1}|\overline{M_2}|\dots|\overline{M_{|D_1|}}|\overline{N_1}|\overline{N_2}|\dots|\overline{N_{|D_2|}}] = |D|$; where the $\overline{M}$'s and $\overline{N}$'s columns are obtained by extending the M's and N's columns to the entire domain D, and the last equality can be deduced for example by performing the Gaussian elimination steps specific to matrices $[M_1|M_2|\dots|M_{D_1}]$ and $[N_1|N_2|\dots|N_{D_2}]$, respectively.

Thus, $\operatorname{rank}\mathcal{OC}(A, B, C_D) = |D|$, which means that within the current network, the set $\{u_1, u_2\}$ is controlling the nodes in $D = D_1 \cup D_2$. □

The correctness of Algorithm 1 follows from Lemma 1. The next theorem analyzes the running time of the algorithm.

Theorem 1 *Given a graph $G = (V, E)$, such that $|V| = n$ and a target set $T \subseteq V$ with $|T| = k$, Algorithm 1 solves the STC problem in time $O(f(k)p(n))$. Thus, the STC problem is fixed parameter tractable.*

Proof: We present in more detail and analyze the running time of each step of Algorithm 1.

Step 1. For each node $v \in V$, we compute and store as follows all the sets of nodes in T that v can simultaneously control. Firstly, we show how to decide if a node $v \in V$ covers a given subset of nodes $T' \subseteq T$ in polynomial time in $|V|$. Given a set of nodes $X \subseteq V$, let $N(X)$ be the open neighborhood of X, that is $N(X) = \{v \in V : \exists a \in X \text{ s.t. } (v, a) \in E\}$. We define the graph $G_{v,T'} = (V', E')$, where:

1. Let $T_0 = T$ and $T_{i+1} = N(T_i)$, $\forall 0 \leq i < n$. The node set V' of the graph $G_{v,T'}$ consists of all the sets T_i plus two other nodes $\{s, t\}$. Since,

the node set V' may contain more copies of the same nodes from V, we refer to a node $p \in V$ that is in the set T_i as p^i. Notice that a node p cannot appear twice in a set T_i.

2. In the edge set E' of the graph $G_{v,T'}$ we add an edge (a^{i+1}, b^i) if $(a, b) \in E$. Moreover, we add an edge between (s, v^i), if $v^i \in T_i$. Finally we add an edge (a, t), $\forall a \in T'$.

The node v can control simultaneously the nodes in the set T' if and only if there exists k'-node disjoint paths from s to t, where $k' = |T'|$. Observe that the graph $G_{v,T'}$ was constructed such that any two node disjoint paths from s to t in $G_{v,T'}$ correspond to paths in G from v to a node in T', paths that do not intersect at the same distance from the nodes in T'. The k nodes disjoint paths problem between two nodes is solvable in time $O(k(n+m))$ on a graph with n nodes and m edges [3]. Thus, since $G_{v,T'}$ has at most n^2 nodes and n^3 edges, finding k disjoint paths between s and t takes time at most kn^3.

Then, to complete Step 1 of Algorithm 1, we repeat the procedure described above for every node $v \in V$ and any subset $T' \subseteq T$. Since there are 2^k subsets of T, the total running time of Step 1 of Algorithm 1 is $O(k2^k n^4)$.

Step 3. Any set C_v has at most k elements and, thus, any set $\mathcal{D}$ has at most 2^k elements. Moreover, to decide if we enter the loop in Step 3, for every set of $\mathcal{D}$ we check if it is one of the sets C_v computed at Step 1. Thus, the complexity of Step 3 of Algorithm 1 is $O(n4^k 2^{2^k})$ times the running time of Steps 3a) and 3b), where the n comes from the running time required to compare two sets of size n.

Notice that since the number of sets in the set cover instance is bounded by 2^k and the number of elements is k, then we can solve the set cover in $O(2^k 2^k) = O(4^k)$, since one can solve Set Cover with set family F and universe U in $O(|F| * 2^{|U|})$ time.

Thus, the overall running time of Algorithm 1 is $O(k2^k n^4 + n4^{2k}2^{2^k})$. □

3.2 Towards Efficient FPT Algorithms Using Multiple Parameters

In recent years, the STC problem has received significant attention in connection to its applicability in bio-medicine and pharmacology, see

e.g. [4, 14, 11]. In this setting, one is required to select a small amount of drugs which, by enabling cascading effects in the protein signaling network, would drive a set of well established key target nodes/proteins to a particular configuration. In turn, this configuration of the target proteins, also known as essential proteins, is expected to correlate with a positive therapeutic effect over the patient. In this setting, the number of internal nodes of the graph G corresponds to the number of proteins within our network, usually in the order of 1000 to 3000. Also, the target T will be given by the set of disease-specific essential proteins present in the network, which in these cases was observed to be in the order of 100 to 200 proteins, i.e., one can roughly assume a 1 to 10 ratio between the number of targets and that of total number of nodes. What is also specific to this setting is that the size of a controlling path, from a driven node to a target, must also be relatively small, i.e., smaller than 10 and preferably around 5. This requirement is due to the fact that such paths translate to cascading effects in the signaling network and, thus, the more intermediary elements within, the less reliable the entire process and the desired outcome becomes.

In the following, we present a fixed parameter tractable algorithm for STC whose time complexity is exponential in the parameters k and p, corresponding to the size of the target set T and the maximum length of the controlling path from a driver to a target node, respectively, and low polynomial in n, the total number of nodes in the network. The algorithm generalizes a Greedy approach first reported in [10] and later analyzed and improved in [4, 14, 11].

Theorem 2 *Given a graph $G = (V, E)$ and a target set $T \subseteq V$ with $|T| = k$ and $|V| = n$, Algorithm 2 solves the Target Controllability Problem in time $O(kn \cdot (\frac{e(n+k)}{k})^{kp})$.*

Proof: In the following, we present in more detail and analyze the running time of each step of Algorithm 2 and of its Control sub-function, i.e., Algorithm 3.

The final controlling set, S_{best}, can be updated only after p nested applications of the iterative Control algorithm. In each of these p nested steps, we need to generate a bipartite graph, enumerate all possible maximal matchings, and form the set S, which will then be fed into the next application of the iterative function Control. While the construction of the bipartite graph can be done in $O(kn)$ time, enumerating all its maximal matchings

Algorithm 2 An FPT algorithm for the STC problem parameterized by k, the size of the target set and p, the maximum length of the controlling path

Input: A directed graph $G = (V, E)$, a set of nodes $T \subseteq V$, $|T| = k$, and an integer p.

Output: A set of nodes $U \subseteq V$ of minimum cardinality that controls T.

1. We create a new graph $G' = (V', E')$. For determining V' we add to V a number of k nodes (denoted $u_1, u_2, \ldots, u_k$) and for E' we add to E a number of k edges, such that the edge $(u_i \to t_i) \in E'$, $\forall 1 \leq i \leq k$.
2. We set $S_{best} = T$ and $S = \emptyset$.
3. We apply the iterative algorithm Control (Algorithm 3) for $(G' = (V', E'), i = 1, T_0 = T, p, S)$.

return S_{best}

requires $O(n)$ per maximal matching, see e.g. [25]. In the worst case scenario, when we are dealing with a complete graph G, all of the intermediary bipartite graphs G_i will also be complete. Thus, in each case, the number of edges will be bounded by $k \cdot (n + k)$ (since we have $|V'| = n + k$ nodes on the left side, and $|T_i| \leq k$ nodes on the right side) while the number of maximal matchings will be upper bounded by $\binom{n+k}{k}$. Therefore, the overall time complexity can be upper bounded by:

$$\underbrace{O(kn + \binom{n+k}{k} \cdot (kn + \binom{n+k}{k} \cdot (\ldots)))}_{p \text{ times}}$$

i.e., $O(\binom{n+k}{k}^p \cdot kn)$. The $(\ldots)$ denote that we have $kn + \binom{n+k}{k}$ nested. As $\binom{n+k}{k} \leq (\frac{e(n+k)}{k})^k$, we get that the running time of the algorithm can be upper bounded by $O(kn \cdot (\frac{e(n+k)}{k})^{kp})$. □

Another sensitive parameter which arises from the applicability of this method in the medical setting comes from restricting the set of nodes from which the control over the target can be initiated, i.e., the set of potential driver nodes. This set corresponds to a medium-sized set of proteins, called drug-targets, which are known to be directly affected (usually down-proliferated) by the use of known drugs. By further specific filtering

Algorithm 3 The iterative function Control called in the main program

Input: A directed graph $G = (V, E)$, an integer i - the current level in the linking graph, a set of nodes T_{i-1} - the current target in the i^{th} level of the linking graph, an integer p - the maximum expansion of the linking graph, and a set of nodes S - the current solution (incomplete if $i < p$),
Output: The set T_i which is the target in the $(i+1)^{\text{th}}$ level of the linking graph and an update of S, the current solution for the driven set. If $i = p$, a possible update of the S_{best} solution.

1. We build a bipartite graph G_i with the nodes in V on the left side (denoted T_i), and the nodes in T_{i-1} on the right side. We add to G_i all of the edges in E that have the source node in T_i and the destination node in T_{i-1}.
2. We enumerate all maximal matchings in the graph G_i between the nodes in T_i and the nodes in T_{i-1}.
3. For each maximal matching, do:
   (a) We remove from T_i all of the nodes left unmatched. We add all unmatched nodes from T_{i-1} to S, if these nodes are not already there.
   (b) (Optionally, to speed up the search, we check if $|S| \geq |S_{best}|$, and if so we backtrack).
   (c) If $i = p$, we add to S all of the nodes in T_i. If $|S| < S_{best}$, then $S_{best} \leftarrow S$.
   (d) If $i \neq p$, we repeat again the iterative algorithm for $(G' = (V', E'), i+1, T_i, p, S)$.

of the types of drugs that the user wants to focus on, the size of this set can be further modified. For example, in [14], the authors use the set of U.S. Food and Drug Administration (FDA) approved drugs, which selects a set of 1500 direct drug-target proteins out of a total of approx 20 000 proteins[7] (excluding post-translational modification and other variants of these, such as phosphorilation, acetilation, etc.). This set can be enlarged

[7]This number comes from the approximate total of 20 000 genes encoded in the human genome.

or restricted by making further choices such as: considering also drugs in clinical trials, experimental drugs, drugs used in oncology, etc. Overall, such a set of potential driver nodes can range from 1/10 to 1/100 of the total set of nodes, which is a substantial restricting parameter. However, limiting the number of potential driver nodes to a subset S of the graph nodes slightly modifies the type of problem we have to study. Indeed, by making this assumption we can no more guarantee that the entire desired target can be controlled. Thus, the Structural Target Controllability Problem becomes a min-max type of question and is defined below. We call this variant of the problem the Driver Restricted STC (DRSTC) Problem.

Definition 4 (Driver Restricted STC Problem (DRSTC)) *What is the minimum number of driver nodes, selected out of the subset S, which can control a maximum number of nodes from T, the target set. Potentially, we can also ask for the specific sets of selected driver nodes and the subsequent controlled nodes.*

In the following, we provide a fixed parameter tractable min-max optimization algorithm for DRSTC, whose time complexity is exponential in the parameters s and p, corresponding to the size of the potential driver set S and the maximum length of the controlling path from a driver to a target node, respectively. Note that from practical purposes, p is a rather small integer, e.g., $p \leq 10$. The algorithm is a further tailored variant of our initial fixed parameter tractable Algorithm 1.

Theorem 3 *Given a graph $G = (V, E)$, a target set $T \subseteq V$ with $|T| = k$, a set of nodes from which the control can be initiated $S \subseteq V$ with $|S| = s$, and an integer p corresponding to the maximal length of a controlling path from a driver to a target node, Algorithm 4 solves the Driver Restricted Target Controllability Problem in time $O(nk^{ps})$, where $n = |V|$.*

Proof: In the following, we present in more details and analyze the running time of each step of Algorithm 4.

In Step 1, for each d_i, each set $T_{d_i}^j \subseteq T, j \leq p$ can contain at most k elements; computing these sets is done in $O(nk^p)$, where $n = |V|$ is the total number of nodes. Since $T_{d_i} \subseteq T$, the maximum number of ways in which we can select T_{d_i} in Step 2 is $k(k-1)(k-2)\ldots(k-p+1) < k^p$, for each $d_i \in S$.

The most computational expensive part of the Algorithm is Step 3, where we have to compute all possible unions of $|S|$ sets, where each set i is

Algorithm 4 An FPT algorithm for the Driver Restricted STC problem parameterized by k, the size of the target set, s, the size of the restricted driver nodes set, and p, the maximum length of the controlling path.

Input: A directed graph $G = (V, E)$, a set of target nodes $T \subseteq V$, $|T| = k$, a set of potential driver nodes $S \subseteq V$, $|S| = s$, and an integer p, the maximum length of a controlling path from a driver to a target node.
Output: A set of nodes $U \subseteq S$ of minimum cardinality that controls a maximum subset of T, i.e., the subset of T controllable from S.

1. For each $d_i \in S, 1 \leq i \leq s$ compute the sets $T^1_{d_i}, T^2_{d_i}, \ldots, T^p_{d_i}$, where $T^j_{d_i} \subseteq T, j \leq p$ contains all those target nodes $t \in T$ from which there exists a directed path of length j from d_i to t, i.e., the nodes from T which are controllable in exactly j steps from d_i.

2. Compute all possible sets T_{d_i} such that exactly one element for each $T^j_{d_i} \subseteq T, j \leq p$ is added to T_{d_i}.

3. Compute all possible unions $\mathcal{T}_S$ for each choices of the sets T_{d_i}, i.e., $\mathcal{T}_S = \{\bigcup_{d_i \in S'} T_{d_i} \mid S' \subseteq S \text{ and } T_{d_i} \text{ computed from above}\}$.

return minimal S' such that there exists T_{d_i}, $d_i \in S'$ such that $\bigcup_{d_i \in S'} T_{d_i}$ is a maximal element of $\mathcal{T}_S$.

either one of the possible choices of T_{d_i} or $\emptyset$, if in that configuration $d_i \notin S'$. Thus, we have to assemble a total of $O((k^p)^s)$ sets.

In order to output the result we have to keep track of the maximal elements of the above sets, as well as the underlying $S' \subseteq S$ which generate them. Thus, the complexity of the algorithm is in $O(nk^{ps})$. □

We mention that at the end of Algorithm 4 we can also output the elements of the set $\bigcup_{d_i \in S'} T_{d_i}$, which represents the subset of target nodes controlled from S'.

Note regarding Algorithm 4: Despite the major reduction of the algorithmic complexity of the STC problem for the restricted case, even moderate sized instances, e.g. Network 3 from Table 1 which has 67 nodes, 14 targets, and 15 potential driver nodes, the algorithm does not end in a reasonable amount of time, i.e. 24 hours on a powerful desktop computer. This tends to suggest that either a considerable improvement needs to be performed to such an exhaustive search algorithm, or the real-case instances of this

problem remains to be tackled only by approximation correspondents.

In Section 5 we introduce a new variation of Algorithm 4 which has a considerable improved complexity, leading it to the possibility of analyzing even real-case instances. The efficiency in the running of the algorithm comes with the drawback that our algorithm is not guaranteed to always return the optimal solution. Nevertheless, we show that our algorithm is almost always optimal.

4 Hardness of Approximation

In this section, we show that the Structural Target Controllability (optimization) problem cannot be approximated within a factor of $(1-\epsilon)\ln k, \forall \epsilon > 0$, unless $NP \subseteq DTIME(n^{\log\log k})$, where k is the number of nodes in the target set T. We prove this via an approximation preserving reduction from the Set Cover problem (see Definition 3). Feige [8] showed that Set Cover is hard to approximate within $(1-\epsilon)\ln k, \forall \epsilon > 0$, unless $NP \subseteq DTIME(k^{\log\log k})$, where k is the number of elements in the universe.

Theorem 4 *Unless $NP \subseteq DTIME(k^{\log\log k})$, the STC problem cannot be approximated within a factor of $(1-\epsilon)\ln k, \forall \epsilon > 0$.*

Proof: Given an instance of the Set Cover problem, i.e., a set $\mathcal{U} = \{u_1, u_2, \ldots, u_k\}$ with k elements and n sets $S_1, S_2, \ldots, S_n \subseteq \mathcal{U}$, we construct the following instance of the STC problem.

1. Add a node $s_i \in V$ corresponding to each set S_i in the Set Cover instance.
2. Add a node $t_i \in V$ corresponding to each element u_i in the set $\mathcal{U}$.
3. For each S_i, add $q_i = |S_i|(|S_i|-1)/2$ *auxiliary* nodes in V. We term these nodes $a^i_1, a^i_2, a^i_3, \ldots, a^i_{q_i}$.
4. The target set T consists of all the nodes $t_i \in V$.
5. For each set S_i of the set cover instance, we construct $|S_i|$ paths of length $2, 3, 4 \ldots |S_i|+1$ as follows. Let $S_i = \{u_1, u_2, \ldots u_{|S_i|}\}$. Then we construct the paths: $\{s_i, t_1\}, \{s_i, a^i_1, t_2\}, \{s_i, a^i_2, a^i_3, t_3\}, \ldots,$ $\{s_i, a^i_{q_i-|S_i|+1}, a^i_{q_i-1}, \ldots, a^i_{q_i}, t_{|S_i|}\}$

We will now show that the Set Cover instance has a solution with x sets if and only if the target set T can be controlled with x driver nodes. Thus, the existence of an approximation algorithm of $(1-\epsilon)\ln k$, for some $\epsilon > 0$, implies the existence of an approximation algorithm with the same factor for the Set Cover problem which implies $NP \subseteq DTIME(n^{\log\log k})$.

Given a Set Cover with x sets $S_{i_1}, S_{i_2}, \dots S_{i_x}$, then the driver nodes $s_{i_1}, s_{i_2}, \dots s_{i_x}$ control all the target nodes since each s_{i_j} controls precisely the target nodes corresponding to the elements in S_{i_j}. This holds since each path from the node s_{i_j} to nodes in T has a different length.

Conversely, given a set of x driver nodes that control all the target nodes, we reconstruct a valid Set Cover with x sets, by choosing the sets corresponding to the driver nodes. Thus, the theorem follows. □

5 An Algebraic Approach for Solving the Driver Restricted Structural Target Controllability Problem

In this section we present a probabilistic heuristic algorithm for the DRSTC problem (Definition 4), algorithm that uses an algebraic approach. As explained in Section 2, the STC problem has an innate algebraic representation (see Definition 1). In the case of a restricting set $S \subseteq X$ of potential driver nodes, the input matrix B is restricted itself by selecting only those nodes from S, i.e., for any $S' \subseteq S$, $S' = \{x_{i_1}, x_{i_2}, \dots, x_{i_m}\}$ we define $B_{S'} \in \mathcal{R}^{n\times m}$ having non-zero values only on the m positions $B(i_j, i_j)$, $i \leq j \leq m$. Thus, given such a restricting set $S \subseteq X$ and a bound $p \leq n$ on maximal length of a controlling path from a driver to a target node, the above algebraic formulation becomes:

Compute a minimal subset $S' \subseteq S$, $|S'| = m$, such that $\operatorname{Srank}\mathcal{OC}_p(A, B_{S'}, C_T) = \operatorname{Srank}\mathcal{OC}_p(A, B_S, C_T)$, where $\mathcal{OC}_p(A, B_{S'}, C_T)$ is the *length p controllability matrix* $\mathcal{OC}_p(A, B_{S'}, C_T) := [C_T B_{S'} \mid C_T A B_{S'} \mid C_T A^2 B_{S'} \mid \dots \mid C_T A^p B_{S'}]$.

As in the case of Algorithm 4, we are going to consider all the subsets S' of S, by eliminating elements from S one-by-one. Then, we will (approximately) compute the generic rank $\operatorname{Srank}\mathcal{OC}_p(A, B_{S'}, C_T)$, and we will compare it with the maximal choice $\operatorname{Srank}\mathcal{OC}_p(A, B_S, C_T)$. The generic rank of a matrix cannot be computed in polynomial time [7]. However, it is known from early works on structural network controllability [21, 29] that

for any LTIS (A, B, C_T), the set $\mathcal{K} = \{(A, B, C_T) \mid \operatorname{rank} \mathcal{OC}(A, B, C_T) = \operatorname{Srank} \mathcal{OC}(A, B, C_T)\}$ is open and dense with respect to operator norm and moreover, more importantly, its complement is of measure zero. That is, the set of values for the non-zero entries in the matrices A, B, and C_T for which $\operatorname{rank} \mathcal{OC}_p(A, B, C_T) < \operatorname{Srank} \mathcal{OC}_p(A, B, C_T)$ is a very sparse set. Thus, for computing the generic rank $\operatorname{Srank} \mathcal{OC}_p(A, B, C_T)$, it is enough to compute the $\operatorname{rank} \mathcal{OC}_p(A, B, C_T)$ (e.g., using the Gaussian Elimination method) for one or several random valuations of the non-zero values in these matrices. Algorithm 5 is a min-max approximation algorithm for DRSTC, whose time complexity is exponential in the parameters s, corresponding to the size of the potential driver set S, times a polynomial in parameters s, p, n and k, corresponding to the maximal length of the controlling path from a driver to a target node, the total number of nodes, and the size of the target set, respectively.

By increasing the number of times the rank computation is repeated, with different random valuations, the algorithm produces a solution closer to the optimum. By the algorithm's design, the algorithm will never output a subset S' which actually does not control a maximal subset of T. Notice that if we were able to compute exactly the generic rank of a matrix, then we can make Algorithm 4 an exact algorithm.

Depending on the level of approximation desired, we choose the constant $Q \geq 1$, as explained in Note 3 below. From practical perspective it is enough to have $Q = 3$. In the next theorem, we show the correctness of Algorithm 5.

Theorem 5 *Given a graph $G = (V, E)$, a target set $T \subseteq V$ with $|T| = k$, a set of nodes from which the control can be initiated $S \subseteq V$ with $|S| = s$, and an integer p corresponding to the maximal length of a controlling path from a driver to a target node, Algorithm 5 produces a feasible (but possibly suboptimal) solution for the Driver Restricted Target Controllability Problem in time $O(2^s \times n^5)$.*

Proof: In the following, we present in more details and analyze the running time of each step of Algorithm 5.

Exploring all possible subsets of S has clearly complexity $2^{|S|}$ (see Note 1 below for a discussion on how to speed up this process in practice).

The next step of the algorithm is to choose Q random valuations for the non-zero values of matrices A, B_S, and C_T: $(A^1, B_S^1, C_T^1) \ldots (A^Q, B_S^Q, C_T^Q)$. For all these valuations compute the rank of $\mathcal{OC}_p(A, B_S, C_T)$ as:

$$\mathcal{OC}_p(A, B_S, C_T) := [C_T B_S \mid C_T A B_S \mid C_T A^2 B_S \mid \ldots \mid C_T A^p B_S] \in \mathcal{R}^{k \times ps},$$

Algorithm 5 A probabilistic heuristic algorithm for the Driver Restricted STC problem parameterized by k, the size of the target set, s, the size of the restricted driver nodes set, and p, the maximal length of the controlling path.

Input: A directed graph $G = (V, E)$, a set of target nodes $T \subseteq V$, $|T| = k$, a set of potential driver nodes $S \subseteq V$, $|S| = s$, and an integer p, the maximal length of a controlling path from a driver to a target node.
Output: A set of nodes $U \subseteq S$ of minimum cardinality that controls a maximal subset of T, i.e., the subset of T controllable from S.

- For each subset $S' \subseteq S$:
  - Compute Q times the rank of $\mathcal{OC}_p(A, B_{S'}, C_T)$ for a random assignment of the non-zero elements of matrices $A, B_{S'}$, and C_T
  - Let Srank'$\,\mathcal{OC}_p(A, B_{S'}, C_T)$ be the maximum of the ranks computed at the previous step

return minimal S' such that Srank'$\,\mathcal{OC}_p(A, B_{S'}, C_T)$ = Srank'$\,\mathcal{OC}_p(A, B_S, C_T)$

and approximate Srank$\,\mathcal{OC}_p(A, B_S, C_T)$ as: Srank'$\,\mathcal{OC}_p(A, B_S, C_T) := \max\{\text{rank}\,\mathcal{OC}_p(A^1, B_S^1, C_T^1), \ldots, \text{rank}\,\mathcal{OC}_p(A^Q, B_S^Q, C_T^Q)\}$.

Computing $\mathcal{OC}_p(A, B_S, C_T)$ for each of the valuations is performed in $O(pn^3)$ time and since p is bounded by n, computing $\mathcal{OC}_p(A, B_S, C_T)$ takes at most $O(n^4)$ time. Also, the rank calculation can be performed using e.g. the Gaussian Elimination method. While this method involves $O(n^3)$ operations, the implementation of the method may create numbers with exponentially many bits. Nevertheless, there is a variant of Gaussian elimination, called the Bareiss algorithm[1], that avoids this exponential growth of the intermediate entries and has a time complexity of $O(n^5)$.

For each of the subsets $S' \subseteq S$ explored above, derive the restricted valuations $B_{S'}^1, \ldots, B_{S'}^Q$ out of the valuations for B_S. Then, as above, compute $\mathcal{OC}_p(A^1, B_{S'}^1, C_T^1)$ and its rank. If $\text{rank}\,\mathcal{OC}_p(A^1, B_{S'}^1, C_T^1) <$ Srank'$\,\mathcal{OC}_p(A, B_S, C_T)$ re-compute the rank for the next valuation. We consider that Srank'$\,\mathcal{OC}_p(A, B_{S'}, C_T) =$ Srank'$\,\mathcal{OC}_p(A, B_S, C_T)$ iff for at least one of the above valuations we have an equality.

In order to output the result we have to keep track of the minimal $S' \subseteq S$ for which Srank'$\,\mathcal{OC}_p(A, B_{S'}, C_T) =$ Srank'$\,\mathcal{OC}_p(A, B_S, C_T)$. Also,

unless this was performed in the initial Gaussian Elimination method implementation, we have to determine which of the lines in $\mathcal{OC}_p(A, B_{S'}, C_T)$ are linearly independent, i.e., which of the targets from T are controlled from S'. Thus, the time complexity of Algorithm 5 is $O(2^s n^5)$. □

5.1 Notes on the Implementation of Algorithm 5

Given the prohibitively high computational complexity of all previous algorithms, we only considered the implementation of Algorithm 5, which is itself a more real-life scenario-oriented version of Algorithm 4. Furthermore, while its computational complexity cannot be improved, we detail below a few algorithmic features which considerably boosted the efficiency of the implementation.

Note 1: This note discusses a Branch and Bound improvement on the Algorithm 5. At the core of Algorithm 5 is a complete subset exploration of S, the collection of potential driver nodes. Moreover, we are interested in finding a minimal $S' \subseteq S$ which can control as much as the entire S. Thus, in our exploration, it makes sense to explore these subsets only as long as they are smaller than the best subset identified so far. In this way the search tree is considerably pruned, significantly improving the run-time of the algorithm.

Note 2: On a similar note, while the algorithm provides a complete subset exploration, there are a number of subsets that can safely be discarded. For example, in the most favorable scenario, each source node would control the maximum of $p+1$ target nodes (itself, and p others), so there must be at least $\lfloor \frac{\text{Srank'}\,\mathcal{OC}_p(A,B_S,C_T)}{p+1} \rfloor$ source nodes in the solution. Similarly, in the least favorable scenario, each target node is controlled by a different source node, so there can be at most $\text{Srank'}\,\mathcal{OC}_p(A, B_S, C_T)$ source nodes in the solution.

Note 3: In our implementation of calculating the generic rank of matrices $\mathcal{OC}(A, B, C)$ we have seldom encountered cases when two different random valuations of the structural matrices A, B, and C would generate a different rank. More importantly, it has never happened that a third random valuation would generate yet another rank value. Thus, in our implementation of Algorithm 5 we have used $Q = 3$.

Note 4: As the algorithm is intended to be applied in the biological domain, where shorter path lengths are to be desired due to the quick dissipation of a drug's effects over long signalling pathways, we used a maximum path length of $p = 5$.

5.2 Results of Algorithm 5

In order to assess the efficiency and the utility of Algorithm 5 for the DRSTC problem, we have run it on several real-life networks of different domains. Firstly, we used the social networks documented in [30] (1) and [18] (2), where the nodes represent people and the edges the positive sentiment from one person to another. Then, we considered the electronic networks presented in [19] (1, 2, 3), where the nodes represent logic gates and the edges the connections between them. Finally, we generated several protein-protein interaction networks starting from the essential genes for breast cancer (1, 3) and pancreatic cancer (2) described in [14], using the OmniPath [6] database, and considering the interactions between the essential genes (1, 2), or the interactions between the essential genes with one intermediary gene (3). For each network, we randomly generated three sets of target nodes (with the sizes of 10, 20 and 30) from the set of nodes with at least one incoming edge. Similarly, for each network, we also randomly generated three sets of source nodes from the set of nodes with at least one outgoing edge. If not enough suitable nodes existed in a network to form a set of certain size, then the corresponding run of the algorithm was skipped.

The results are presented in Table 1, and are available, together with the implementation and the data sets, at [23]. In Table 1, the last three columns are, CT: number of controlled target nodes; CS: number of controlling source nodes; TS: running time, in seconds.

The runs that did not complete within two days have been omitted / marked with $*$. As it can be seen in Table 1, Algorithm 5 can be successfully applied on real-life networks.

As expected, the parameters with the biggest influence on the running time are the size of the set of source nodes and the number of nodes in the network. While the total running time of the algorithm mostly depends on the total number of subset (and, thus, on the size of the set of source nodes) and was significantly decreased by skipping the invalid subsets and not taking into consideration the larger solutions once a smaller one was found, the running time per subset is dependent on the rank computations (which, in turn, depends on the number of nodes in the network, and to a lesser extent on the size of the sets of nodes).

Furthermore, this strategy helps the algorithm perform faster on well-connected networks, where few source nodes can control the target set of nodes and, thus, are quickly found. This can be observed in the case of the largest protein-protein interaction network (which is the most well-connected

Network	Nodes	Edges	Targets	Sources	CT	CS	TS
Social Interaction 1	32	96	10	10	10	2	0.11
			10	20	10	2	2.5
			10	30	10	2	2863.6
			20	10	20	4	0.23
			20	20	20	4	2.39
			20	30	20	4	2886.38
Social Interaction 2	67	182	10	10	10	3	0.22
			10	20	10	2	2.43
			10	30	10	2	5766.48
			20	10	20	4	0.3
			20	20	20	4	2.95
			30	10	30	6	0.53
			30	20	29	5	3.4
Electronic Circuit 1	122	189	10	10	3	3	0.28
			10	20	8	3	73.62
			10	30	9	5	29228.78
			20	10	10	6	2.9
			20	20	18	6	284.15
			30	10	13	5	2.63
			30	20	25	8	655.04
Electronic Circuit 2	208	189	10	10	3	3	1.21
			10	20	3	3	11.62
			10	30	6	3	8764.32
			20	10	5	5	5.46
			20	20	5	4	120.75
			30	10	7	7	12.76
			30	20	3	3	6.08
Electronic Circuit 3	512	819	10	10	4	3	10.53
			10	20	3	2	26.15
			20	10	10	6	150.67
			20	20	8	6	5482.52
			30	10	12	7	158.38
			30	20	12	6	106267.06
Protein-Protein Interaction 1	64	94	10	10	7	3	0.19
			10	20	8	3	2.73
			10	30	10	4	6870.62
			20	10	12	4	0.32
			20	20	18	8	387.45
			30	10	18	4	0.31
			30	20	28	11	843.31
Protein-Protein Interaction 2	58	64	10	10	7	3	0.18
			10	20	10	4	6.07
			10	30	10	4	3951.35
			20	10	8	4	0.35
			20	20	11	6	390.33
			30	10	13	4	0.37
			30	20	20	10	1041.5
Protein-Protein Interaction 3	433	1604	10	10	10	3	5.85
			10	20	10	3	17.89
			10	30	10	3	2921.13
			20	10	18	4	11.68
			20	20	18	4	108.76
			30	10	29	7	42.74
			30	20	28	6	1474.74

Table 1: Results of Algorithm 5.

of the analyzed networks), on which the algorithm completed significantly faster for all sets compared to the largest similarly-sized electronic circuit (which is less-connected).

The order of the nodes within the sets of source nodes also has a large influence on the final running time. For example, if the last node in the list would be required for the best solution, then the said best solution can only be found within the last half of the checked subsets (based on the order

the algorithm generates and iterates through them), while the algorithm would still check all previous subsets, resulting in a significant increase of the running time.

However, the final running time could be further improved by decreasing the number of times that the rank of the corresponding matrices is computed, further shortening the maximum path length, or by using a different order (and stopping condition) for analyzing the subsets.

6 Conclusions and Future Work

Network Science has been proven to be highly relevant within the current developments of medicine and personalized therapeutics. Within this field, structural network control is a powerful and efficient tool for steering the involved bio-medical systems towards desirable configurations. Thus, the algorithmic optimization problems studied in this manuscript are relevant for the computational bio-medicine community, as highly optimized solutions have a significant chance of translating into efficient therapeutics. Although the Structural Target Control (Optimization) problem has been proven to be NP-hard in its general case and can not even be approximated within a constant factor, and although it is a known fact that bio-medical networks are rather large, containing thousands of nodes and (tens of thousands of) interactions, in practice, several of the involved parameters can still be considerably bounded to significantly lower values. In this research, we took advantage of these insights in order to provide several optimization algorithms which remain of low polynomial complexity with regards to the size of the network, and are exponential only in those chosen parameters. Out of these algorithms, one in particular has been shown to be tractable for real-case networks containing up to 200+ nodes. Moreover, on all non-trivial test-cases, this latter algorithm gave more detailed and complete output than the current state of the art software dealing with this problem.

Acknowledgments

This work was supported by the Academy of Finland through grant 272451, by the Finnish Funding Agency for Innovation through grant 1758/31/2016, by the Romanian National Authority for Scientific Research and Innovation, through the POC grant P_37_257 and by a grant of the Romanian Ministry of

Research, Innovation and Digitization, CNCS - UEFISCDI, project number PN-III-P1-1.1-TE-2021-0253, within PNCDI III.

References

[1] Erwin H. Bareiss. Sylvester's identity and multistep integer-preserving Gaussian elimination. *Mathematics of Computation*, 22(103):565–578, 1968. doi:10.1090/S0025-5718-1968-0226829-0.

[2] Vincent A. Blomen, Peter Májek, Lucas T. Jae, Johannes W. Bigenzahn, Joppe Nieuwenhuis, Jacqueline Staring, Roberto Sacco, Ferdy R. van Diemen, Nadine Olk, Alexey Stukalov, Caleb Marceau, Hans Janssen, Jan E. Carette, Keiryn L. Bennett, Jacques Colinge, Giulio Superti-Furga, and Thijn R. Brummelkamp. Gene essentiality and synthetic lethality in haploid human cells. *Science*, 350(6264):1092–1096, 2015. doi:10.1126/science.aac7557.

[3] J. Adrian Bondy and Uppaluri S. R. Murty. *Graph Theory*. Graduate Texts in Mathematics. Springer, 2008. doi:10.1007/978-1-84628-970-5.

[4] Eugen Czeizler, Wu Kai Chiu, Cristian Gratie, Krishna Kanhaiya, and Ion Petre. Structural target controllability of linear networks. *IEEE ACM Transactions on Computational Biology and Bioinformatics*, 15(4):1217–1228, 2018. doi:10.1109/TCBB.2018.2797271.

[5] Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999. doi:10.1007/978-1-4612-0515-9.

[6] Julio Saez-Rodriguez Dénes Türei, Tamás Korcsmáros. Omnipath: guidelines and gateway for literature-curated signaling pathway resources. *Nature Methods*, 12(13):966–967, 2016. doi:10.1038/nmeth.4077.

[7] Björn Engquist. *Encyclopedia of Applied and Computational Mathematics*. Springer Publishing Company, Incorporated, 1st edition, 2016. doi:10.1007/978-3-540-70529-1.

[8] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998. doi:10.1145/285055.285059.

[9] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-29953-X.

[10] Jianxi Gao, Yang-Yu Liu, Raissa M. D'Souza, and Albert-László Barabási. Target control of complex networks. *Nature Communications*, 5:5415, 2014. doi:10.1038/ncomms6415.

[11] Wei-Feng Guo et al. A novel algorithm for finding optimal driver nodes to target control complex networks and its applications for drug targets identification. *BMC Genomics*, 19(1):924, 2018. doi:10.1186/s12864-017-4332-z.

[12] Paul Hines, Seth Blumsack, Eduardo Cotilla Sanchez, and Clayton Barrows. The topological and electrical structure of power grids. In *Proceedings of 43rd Hawaii International International Conference on Systems Science (HICSS-43 2010),*, pages 1–10. IEEE Computer Society, 2010. doi:10.1109/HICSS.2010.398.

[13] Rudolf E. Kalman. Mathematical description of linear dynamical systems. *Journal of the Society for Industrial and Applied Mathematics Series A Control*, 1(2):152–192, jan 1963. doi:10.1137/0301010.

[14] Krishna Kanhaiya, Eugen Czeizler, Cristian Gratie, and Ion Petre. Controlling directed protein interaction networks in cancer. *Scientific Reports*, 7(1):10327, 2017. doi:10.1038/s41598-017-10491-y.

[15] A. Li, S. P. Cornelius, Y.-Y. Liu, L. Wang, and A.-L. Barabási. The fundamental advantages of temporal networks. *Science*, 358(6366):1042–1046, 2017. doi:10.1126/science.aai7488.

[16] Ching-Tai Lin. Structural controllability. *IEEE Transactions on Automatic Control*, 19(3):201–208, 1974. doi:10.1109/TAC.1974.1100557.

[17] Yang-Yu Liu, Jean-Jacques Slotine, and Albert-László Barabási. Controllability of complex networks. *Nature*, 473(7346):167–173, 2011. doi:10.1038/nature10011.

[18] Duncan MacRae. Direct factor analysis of sociometric data. *Sociometry*, 23(4):360–371, 1960. URL: http://www.jstor.org/stable/2785690.

[19] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002. doi:10.1126/science.298.5594.824.

[20] Kazuo Murota. *Systems analysis by graphs and matroids: structural solvability and controllability*. Algorithms and combinatorics. Springer-Verlag, 1987. doi:10.1007/978-3-642-61586-3.

[21] Kazuo Murota and Svatopluk Poljak. *Note on a Graph-theoretic Criterion for Structural Output Controllability*. KAM series, discrete mathematics and combinatorics, operations research, mathematical linguistics. Department of Applied Mathematics, Charles University, 1989.

[22] Svatopluk Poljak. On the generic dimension of controllable subspaces. *IEEE Transactions on Automatic Control*, 35(3):367–369, 1990. doi:10.1109/9.50361.

[23] Victor Popescu. Fixed parameter algorithms for structural target controllability. https://github.com/vicbgdn/FixParamAlgNetControl, 2020.

[24] Robert W. Shields and J. Boyd Pearson. Structural controllability of multiinput linear systems. *IEEE Transactions on Automatic Control*, 21(2):203–212, apr 1976. doi:10.1109/TAC.1976.1101198.

[25] Takeaki Uno. Algorithms for enumerating all perfect, maximum and maximal matchings in bipartite graphs. In Hon Wai Leong, Hiroshi Imai, and Sanjay Jain, editors, *Proceedings of 8th International Symposium on Algorithms and Computation, ISAAC '97*, volume 1350 of *Lecture Notes in Computer Science*, pages 92–101. Springer, 1997. doi:10.1007/3-540-63890-3_11.

[26] Vijay V. Vazirani. *Approximation algorithms*. Springer, 2001. doi:10.1007/978-3-662-04565-7.

[27] Arunachalam Vinayagam, Travis E. Gibson, Ho-Joon Lee, Bahar Yilmazel, Charles Roesel, Yanhui Hu, Young Kwon, Amitabh Sharma, Yang-Yu Liu, Norbert Perrimon, and Albert-Lãszlô Barabâsi. Controllability analysis of the directed human protein interaction network identifies disease genes and drug targets. *Proceedings of the National Academy of Sciences*, 113(18):4976–4981, 2016. doi:10.1073/pnas.1603992113.

[28] Tim Wang, Kıvanç Birsoy, Nicholas W. Hughes, Kevin M. Krupczak, Yorick Post amd Jenny J. Wei, Eric S. Lander, and David M. Sabatini. Identification and characterization of essential genes in the human genome. *Science*, 350(6264):1096–1101, 2015. doi:10.1126/science.aac7041.

[29] Jacques L. Willems. Structural controllability and observability. *Systems and Control Letters*, 8(1):5–12, oct 1986. doi:10.1016/0167-6911(86)90023-X.

[30] Leslie D. Zeleny. Adaptation of research findings in social leadership to college classroom procedures. *Sociometry*, 13(4):314–328, 1950. doi:10.2307/2785274.

[31] Tianzuo Zhan and Michael Boutros. Towards a compendium of essential genes – from model organisms to synthetic lethality in cancer cells. *Critical Reviews in Biochemistry and Molecular Biology*, 51(2):74–85, 2016. PMID: 26627871. doi:10.3109/10409238.2015.1117053.