A Polynomial-Time Deterministic Algorithm for An NP-Complete Problem

Xinwen Jiang* Holden Wool†

Dec 2, 2024

Abstract

An NP-complete graph decision problem, the "*Multi-stage graph Simple Path*" (abbr. MSP) problem, is introduced. The problem is about the decision of existence of specific "global paths" in a graph G. We show that the MSP problem can be solved in polynomial ($O(|E|^9)$)) time, by proposing a poly-time graph algorithm and the proof of its correctness. Our result implies NP = P, and hence no chance is left for NP ≠ P any more.

The algorithm exploits the data structure of *reachable-path edge-set* $R(e)$. By establishing the inter-play between preceding decisions and subsequent decisions, the computed (in a monotonically decreasing manner) information for $R(e)$ carries all necessary contextual information, and can be utilized to summarize the "history" and to detect the "future" for searching "global paths". Paths are always regarded as a collection of edge sets, for the avoidance of exponential complexity. Our proof of the algorithm builds upon a mathematical inductive proving framework, which relies on a crucial structural property of the MSP problem: all MSP instances are arranged into the sequence $\{G_0, G_1, G_2, \ldots\}$, and each G_j $(j > 0)$ in the sequence must have some G_i $(0 \leq i < j)$ that keeps completely accordant with G_j on the existence of "global paths".

* Corresponding author. Partially supported by the National Natural Science Foundation of China ("*Research on the Complexity to Solve An NPC Problem*", No. 61272010). Email: xinwenjiang@sina.com and xinwenjiang@xtu.edu.cn

† Email: holdenwool@foxmail.com

Contents

1 Introduction

The community has made great efforts [Wog22] on the long-standing well-known P vs. NP problem [GJ79, Coo03].

As categorized by Lance Fortnow [For09, For21], a number of techniques—e.g., diagonalization [Tur36, Can74, TJR75, Mel07], circuit complexity [FSS84, Raz85, Raz89, RR97], proof complexity [Hak85], and algebraic geometry [MS01, BI11, Mul12]—have been borrowed or proposed to prove NP $\neq$ P and other related problems.

Efforts on NP = P are mostly engaged in the search of poly-time algorithms for NP-complete problems. Successive successful algorithms for hard problems (e.g., the AKS algorithm for Primality Test [AKS04], the quasipoly-time algorithm for Graph Isomorphism [Bab16], the holographic algorithm for counting problems [Val02] and the many constructive disproofs of popular conjectures in cryptography [Vio18]) have proved repeatedly that people have grossly underestimated the reach of efficient computation in a variety of contexts and thus inspired such efforts. Don Knuth [Knu02] believes that NP = P, but also believes that even if a proof was given, it might not be constructive; or even if an algorithm could be found, it would be too complex to be of practical significance.

First introduced in [JPW10], the "*Multi-stage graph Simple Path*" (abbr. MSP) problem was shown to be poly-time reducible from the famous NP-complete *Hamilton Circuit* (abbr. HC) problem. Ten years later, a paper in Chinese version [Jia20] was published in July 2020, in which a poly-time algorithm for the MSP problem was presented. This has caused widespread concerns and huge amount of discussions.

This paper is directly focused on the study of a poly-time graph algorithm (the *ZH algorithm*) for the NP-complete MSP problem, which further greatly simplifies and refines its sufficiency proof given in [Jia20]. For sake of being self-contained, of the problems caused by different languages and of the convenience of reading, we will include the formal definitions of the MSP problem and the ZH algorithm which were given in [Jia20]. The major work of the current paper is a significantly simplified and refined proving framework of the sufficiency of the ZH algorithm, together with a rigorous and complete proof:

(1) Simplification of the induction variable $f(G)$. The right-hand addition operand L is dropped from the original $f(G) = \left(\sum_{v\in V-\{S,D\}} (d^-(v)-1)\right) + L$, so that our mathematical induction on $f(G)$ can be just done by a split transformation for reducing $\sum_{v\in V-\{S,D\}} (d^-(v)-1)$, without another compact transformation for further reducing L.

(2) Restriction to a more specific problem called $2-$MSP. In-degrees are limited within 2 and out-degrees never exceeds in-degrees. Hence, during the split transformation, the indeterminate discussion of "$x+y=z$" $(x,y,z\in\{1,2,\ldots\})$ becomes the accurate one of "$1+1=2$", and a situation of clear-cut "either-or" logic thinking can be formed.

(3) Aanalytical definitions of basic operators and justification of their consistency with the original procedure definitions. Procedure forms are easier for analyzing and reducing computational complexity, which is the ultimate goal of the paper; while analytical forms better help describe the mathematical properties of basic operators.

(4) Thorough proof of the correctness of the ZH algorithm and examination of it to detail. Our proof provides a sound mathematical foundation for our claim of NP = P.

The motivation of our focus on MSP is that we discover a rich and metrizable structural property of the problem, which naturally gives rise to a proving framework of mathematical induction. Henceforth, designing a poly-time algorithm that can fulfill the proving framework becomes our pursuit. Meanwhile, that property and the proving framework also make it possible

for the accurate proof of the correctness of the algorithm.

Online tutorials of the paper are available at *https://tcsrepositories.github.io/PvsNP/*, *https://weibo.com/p/1005051423845304*.

1.1 Overview of our results

This paper introduces a novel graph decision problem—the MSP problem, together with its restricted form called $2-$MSP. We prove that:

Theorem 1 (NP-completeness). $2-MSP \in NPC$.

A poly-time graph algorithm named the ZH algorithm for the $2-$MSP problem is proposed, which is composed of four basic operators (Operator 1,2,3,4). Given the definitions of the basic operators, we show that:

Theorem 2 (Equivalence & uniqueness, Operator 3, informal). *Operator 3 can be uniquely computed by our definition.*

Theorem 3 (Equivalence & uniqueness, Operator 4, informal). *Operator 4 can be uniquely computed by our definition.*

Let E be the number of edges of the graph G in a given $2-$MSP problem, it's proved that:

Theorem 4 (The cost). *The cost of the ZH algorithm can be* $O(|E|^9)$.

The correctness of the ZH algorithm consists of the necessity and the sufficiency. It's easy to prove the direction of necessity:

Theorem 5 (The necessity, informal). *If G contains a σ-path, then the established decision condition by the ZH algorithm is true.*

For the direction of sufficiency, the ZH algorithm is then embedded in a specially constructed algorithm called the Proving algorithm (abbr. PA).

A linear-order metric $f(G)$ is devised based on the structure property of the MSP problem, to arrange the problem instances into a sequence for performing inductive proof. It's guaranteed by definition that $f(G) \geq 0$. We can first prove the following inductive basis for PA:

Lemma 1 (Informal). *For any G, if $f(G) = 0$, the PA can make a correct decision for G.*

Then, assuming that the PA can make a correct decision for any G' that $f(G') < m$ $(m > 0)$, we can prove that:

Lemma 2 (Informal, the major hard stone). *For any G, if $f(G) = m$, the PA can make a correct decision for G.*

Summarizing Lemma 1,2, we can obtain:

The αβ lemma (Summarizing Lemma 1,2, informal). *The PA can make a correct decision for the graph in any $2-$MSP problem.*

Subsequently, by leveraging the $\alpha\beta$ lemma, the sufficiency of the ZH algorithm can be finally proved:

Theorem 6 (The sufficiency, informal). *If the established decision condition by the ZH algorithm is true, then G contains a σ-path.*

Combining Theorem 1,4,5,6, we can eventually obtain the following theorem:

Theorem 7 ($\mathbf{NP = P}$). *There exists a poly-time algorithm for $2-MSP$, i.e., there exists a poly-time algorithm for NP-complete problems.*

1.2 Technical contributions

The main contribution of the paper is fourfold:

(1) On the MSP problem structure.

The MSP problem is about the decision of existence of specific "global paths" in a graph G. A crucial structural property of the MSP problem is discovered, whereby all MSP instances are arranged into the sequence $G_0, G_1, G_2, \ldots$ (G_k essentially stands for a group of graphs for all $k \geq 0$). For each G_j $(j > 0)$ in the sequence, there is a graph G_i $(0 \leq i < j)$ mathematically equivalent to G_j which keeps completely accordant with G_j on the existence of "global paths". This naturally provides a chance of applying mathematical induction for the proof of an algorithm. In previous attempts, algorithms used for making global decisions were mostly heuristic and intuitive. The ZH algorithm is not guided by intuition; rather, it is dedicatedly designed to comply with the proposed proving framework of mathematical induction. This approach practices the law that, "*one can only rely on logical reasoning whenever intuition fails*".

(2) On the tackling of the complexity.

Although the ZH algorithm deals with paths, it always regards paths as a collection of edge sets. This is the key to the avoidance of exponential complexity.

(3) The data structure of *reachable-path edge-set* $R(e)$ and the computation for $R(e)$.

When solving NP-complete problems, any poly-time algorithm that seeks global information can barely avoid the error caused by localized computation. In the ZH algorithm, the proposed reachable-path edge-set $R(e)$ and the computed information for it carry all necessary contextual information, which can be utilized to summarize the "history" and to detect the "future" for searching "global paths".

(4) The discovery of the relation between local and global strategies.

The relation between local strategies and global strategies is discovered and established, wherein preceding decisions can pose constraints to subsequent decisions (and vice versa). This interplay exploited resembles the paradigm of dynamic programming, while being much more convoluted. Nevertheless, the computation is always strait forward and decreases monotonically (hence invariably finishes in polynomial time), until the emergence of the ultimate decision.

The combination of the above (1)-(4) is just the key towards the overcome of the long-standing complexity barrier.

More insights and brief reviews of related works are available in Section 4, therein why the complexity barrier can get overcome by our method is explained in depth.

2. The problems of MSP, $2-$MSP

Definition 1 (Labeled multi-stage graph). A *labeled multi-stage graph* $G =< V, E, S, D, L, \lambda >$ is a special directed acyclic graph (DAG), where:

- V is the set of vertices, which is divided into $L+1$ ($L \geq 5$ is required by us) *stages*: $V = \bigcup_{0 \leq l^{[1]} \leq L} V_l$ ($V_i \cap V_j = \emptyset$, $0 \leq i \leq L$, $0 \leq j \leq L$, $i \neq j$). A vertex u is *a vertex of stage* l, if $u \in V_l$ ($0 \leq l \leq L$).
- V_0 consists of a single vertex, i.e., the *source* S. V_L consists of a single vertex, i.e., the *sink* D.
- E is the set of edges. Each edge is given as $\langle u, v, l \rangle$ ($u \in V_{l-1}$,$v \in V_l$,$1 \leq l \leq L$) instead of the conventional 2-tuple form, which is called *an edge of stage* l. We use $d^-(v)$ and $d^+(v)$ to each denote the in-degree and out-degree of v.
- λ is a mapping from $V - \{S\}$ to 2^E. $\lambda(v)$ ($v \in V - \{S\}$, $\lambda(v) \subseteq E$) is called the *label of* v.

Definition 2 (ω-path, σ-path). Let $G =< V, E, S, D, L, \lambda >$ be a labeled multi-stage graph. (1) If $P = a - \cdots - b \subseteq E$ such that $P' \subseteq \lambda(v)$ for each $P' = a - \cdots - v \subseteq P$ (note that a path is essentially a set of edges), then P is called a *weak simple path*[2] (abbr. *ω-path*). (2) If $P = S - \cdots - D \subseteq E$ such that $P' \subseteq \lambda(v)$ for each $P' = S - \cdots - v \subseteq P$, then P is called a *simple path* (abbr. *σ-path*).

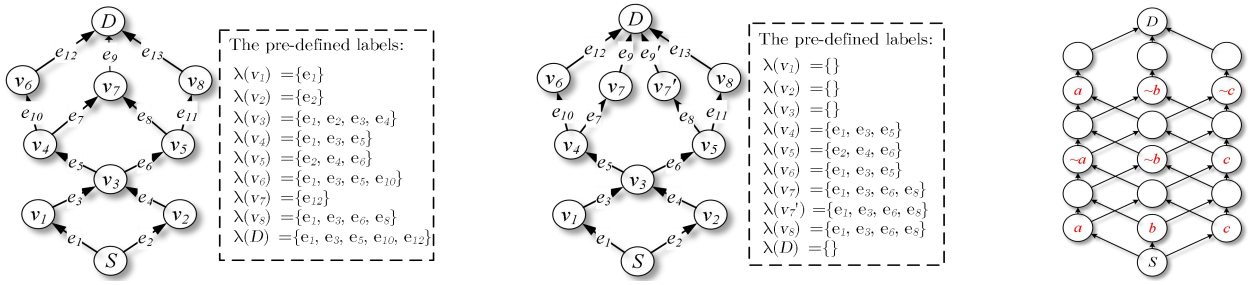


Figure 1: Labeled multi-stage graphs

Definition 3(a) (The "Multi-stage graph Simple Path" problem, abbr. MSP). The *MSP problem* asks whether a given labeled multi-stage graph $G =< V, E, S, D, L, \lambda >$ contains a σ-path.

The MSP instance illustrated in Figure 1(a) contains σ-paths (e.g., $S - v_1 - v_3 - v_4 - v_6 - D$) and ω-paths (e.g., $v_1 - v_3 - v_4$, $S - v_2 - v_3 - v_5$), while the one in Figure 1(b) contains no σ-path. The existence of σ-paths in a graph depends on its structure, as well as its labels.

For technical reasons, we will further focus on a restricted form of MSP, as follows.

Definition 3(b) ($2-$MSP). The $2-$*MSP problem* is a special MSP problem. The basic

[1] Indices are in $\mathbb{N}$ (the set of nature numbers, including zero) by default.

[2] Get distinguished from the conventional concept of "simple path" in graph theory. The latter only requires the path to traverse a vertex no more than once, which is always satisfied in a DAG. However, edges on a path might be rejected by labels on the path, to describe which we borrow the term "simple".

structure of $2-\text{MSP}$ is as shown in Figure 1(c), generally: for each vertex, its in-degree and out-degree are within 2; for each vertex of stage $L-1$, its in-degree equals to 1. The $3-\text{SAT}$ problem can be polynomially converted to the $2-\text{MSP}$ problem. In the inductive proof of our algorithm, we have to construct a pair of logically equivalent graphs. We list several key structural properties item by item in the definition of $2-\text{MSP}$, so as to check the properties of the constructed graph against the original graph one by one, and to also facilitate the adaptation of the changes inevitably caused by us to the constructed graph. To be exact, the properties of items we require the graph $G =< V, E, S, D, L, \lambda >$ in $2-\text{MSP}$ to fulfill are:

(1) $d^+(v) > 0$ $(v \in V - \{D\})$; $d^-(v) > 0$ $(v \in V - \{S\})$. *(Each vertex should appear on some path $S - \cdots - D \subseteq E$.)*

(2) $d^-(v) \leq 2$ $(v \in V - \{S, D\})$; $d^-(v) = 1$ $(v \in V_{L-1})$. *(In-degrees are limited.)*

(3) $\left(\forall v \in \bigcup_{1<i<L} V_i\right) \left((d^-(v) \leq 1) \Rightarrow \left(\begin{array}{c} \forall \langle a, b, h \rangle \in \\ (v - \cdots - D) \subseteq E \end{array} \right) (d^-(a) \leq 1) \right)$. *(Roughly, if a vertex is not multi-in-degree, then neither is any vertex except D on subsequent paths.)*

(4) $\left(\forall v \in \bigcup_{1<i<L-2} V_i\right) (d^+(v) \leq d^-(v))$.

(5) $\lambda(D) = E$.

Theorem 1 (NP-completeness). $2 - MSP \in NPC$. (trivial; proved in Section 5)

The problems of MSP and $2-\text{MSP}$ properly provide a "split"-based inductive proving framework towards the resolution of the P vs. NP problem. Details of the motivation and the proving framework are each discussed in Section 4.1 and Section 3.3.

3 The ZH algorithm for $2-\text{MSP}$

3.1 Basic operators

The ZH algorithm utilizes four basic operators on edge sets for a given $G =< V, E, S, D, L, \lambda >$, as follows. Several observations are provided in Section 6 to help grasp them.

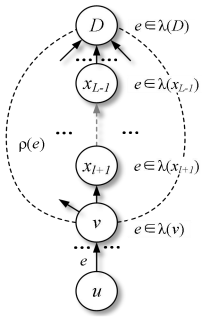


Figure 2: $\boldsymbol{\rho(e)}$

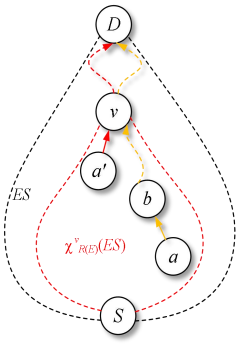
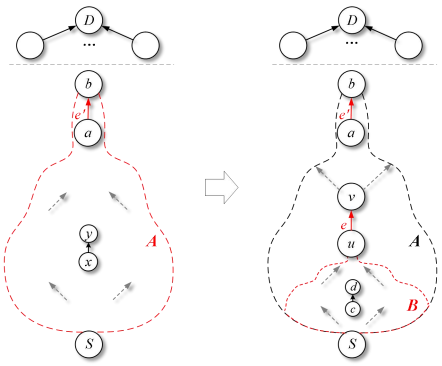


Figure 3: $\boldsymbol{\chi^{v}_{R(E)}(ES)}$



(a) (b)

Figure 4: $\boldsymbol{\psi_{R(E)-\{R(e)\}}(R(e))}$

Operator 1 ($[ES]_u^v$). Given $ES \subseteq E, \{u, v\} \subseteq V$. $[ES]_u^v =_{\text{def}} \{e | e$ is on some path[3] $u - \cdots - v \subseteq ES\}$.

The operator extracts the edges of paths between two designated vertices for a certain collection of edges. The CONNECTIVITY problem is known to be solvable in $O(|E|)$.

Operator 2 ($\rho(e)$). Given $e = \langle u, v, l\rangle \in E$. $\rho(e) =_{\text{def}} [\{\langle a, b, k\rangle \in E \mid e \in \lambda(a) \cap \lambda(b)\}]_v^D$.

The operator captures a necessary condition of σ-path existence. By definition, $\rho(e)$ collects the edges on every $v - x_{l+1} - \cdots - x_{L-1} - D \subseteq E$, if $e \in \lambda(v) \cap \lambda(x_{l+1}) \cap \cdots \cap \lambda(x_{L-1}) \cap \lambda(D)$, as illustrated by the region enclosed by dotted curves in Figure 2. As can be hence observed, for each $\langle x_{i-1}, x_i, i\rangle$ $(1 \le i \le L)$ on a σ-path $P = x_0 - x_1 - \cdots - x_L$ $(x_0 = S,\ x_L = D)$, we have $\langle x_{i-1}, x_i, i\rangle \in \bigcap_{i \le j \le L} \lambda(x_j)$ and thus $\rho(\langle x_{i-1}, x_i, i\rangle) \supseteq [P]_{x_i}^D$.

The cost of $\rho(e)$ can be $O(|E|)$.

We use the notation $R(e)$ (i.e., $R(\langle u, v, l\rangle)$) as a global variable, which initially holds the result of $\rho(e)$ and will be updated later by the ZH algorithm. $R(e)$ carries all contextual information needed by e to detect the "future" (i.e., the containment of e by labels) for searching σ-paths. Let's denote $R(E) = \{R(e) | e \in E\}$.

Definition 4 (ρ-path, reachability). Each path $v - \cdots - D \subseteq R(e)$ $(e = \langle u, v, l\rangle \in E)$ is called a *reachable path* (abbr. *ρ-path*) *of* e. $R(e)$ is called the *ρ-path edge-set of* e, which characterizes the *reachability* of e during the computation of the ZH algorithm. $R(E)$ is called the *collection of ρ-path edge-sets*.

Operator 3 ($\chi_{R(E)}^v(ES)$, procedural form). Given $ES \subseteq E$, $v \in V_l$ and the collection of ρ-path edge-sets $R(E)$. The result of Operator 3, given as the following "procedural form", equals to the final stable ES':

(1) $ES' \leftarrow ES$

(2) **for** $e = \langle a, b, k\rangle \in ES'$
  if $[R(e) \cap ES']_b^v = \emptyset$ $(k < l)$
    then $ES' \leftarrow ES' - \{e\}$
  if $[R(e)]_v^D = \emptyset$ $(k = l \neq L)$
    then $ES' \leftarrow ES' - \{e\}$

(3) $ES' \leftarrow [ES']_S^v$

(4) **repeat** (2),(3) **until** ES' becomes stable

The operator utilizes the ρ-path edge sets in $R(E)$ to compact the whole set ES, as illustrated by the innermost region enclosed by dotted curves in Figure 3. Intuitively speaking, the compacted set ES' is the collection of connected edges $e = \langle a, b, k\rangle \in ES$, such that some ρ-path $P \subseteq R(e)$ should "cling" onto the edges in the compacted set to "climb" towards v (i.e., $[P]_b^v \subseteq ES'$).

It should be noted that, b can be v or D. To maintain the intended semantics of the operator, for these boundary conditions, the definition is slightly different. When $b = v$ $(v \neq D)$, the pruning of e is decided on the content of $[R(e)]_v^D$, instead of $[R(e) \cap ES']_b^v$. When $b = v$ $(v =$

[3] When discussing connectivity, such paths are taken as a whole set of edges (via polynomial-time connectivity check), rather than being distinguished from each other (via exponential-time path enumeration). The same is with the below.

D), we always have $R(e) = \emptyset$ and thus we shall never prune e simply by the content of $R(e)$.

The result of Operator 3 is uniquely determined, regardless of the order of choice of the edges to be pruned during the iteration (see Theorem 2 in Section 7).

Step (2),(3) can be done in $O(|E|^2)$. The execution can terminate within $|E|$ iterations, since at least one edge is pruned per round. Thus, the overall cost is $O(|E|^3)$.

Operator 4 ($\psi_{\boldsymbol{R(E)-\{R(e)\}}}(\boldsymbol{R(e)})$, procedural form). Given $e = \langle u, v, l\rangle \in E$ $(1 < l < L)$ and the collection of ρ-path edge-sets $R(E)$. Operator 4 uses $R(E) - \{R(e)\}$ to restrain $R(e)$, given as the following "procedural form":

(1) **for** $e' = \langle a, b, k\rangle \in R(e)$ (from $k = l + 1$ to $k = L$)
$\boldsymbol{A} \leftarrow \chi^{b}_{R(E)}(\{\langle x, y, i\rangle \in E | e' \in [R(\langle x, y, i\rangle) \cap \lambda(b)]^{b}_{y}\} \cup \{e'\})$
$\boldsymbol{B} \leftarrow \chi^{u}_{R(E)}(\{\langle c, d, j\rangle \in \boldsymbol{A} | \{e, e'\} \subseteq [R(\langle c, d, j\rangle) \cap \boldsymbol{A}]^{b}_{d}\})$
if $\boldsymbol{B} = \emptyset$[4]
then $R(e) \leftarrow R(e) - \{e'\}$

(2) $R(e) \leftarrow [R(e)]^{D}_{v}$

(3) **repeat** (1),(2) **until** $R(e)$ becomes stable

The result of the above definition is uniquely determined, regardless of the order of choice of the edges to be pruned during the iteration (see the following Theorem 3).

Note that, the $R(e)$ modified by Operator 4 now becomes a subset of the original $R(e)$, but we would rather still call each $v - \cdots - D \subseteq R(e)$ a *ρ-path of e* and call $R(e)$ *the ρ-path edge-set of e*. $\psi_{R(E)-\{R(e)\}}(R(e))$ utilizes $(R(E) - \{R(e)\})$ to restrict each $e' \in R(e)$, thus "binding" related ρ-path edge-sets all together.

It will be seen later that, Operator 4 is going to be used iteratively by the ZH algorithm to prune $R(e) \in R(E)$, until each $R(e) \in R(E)$ becomes stable; the computation is always strait forward and decreases monotonically. This technique lies in the center of the ZH algorithm, which realizes the exploitation of the relation between local strategies and global strategies. This resembles the paradigm of dynamic programming, nevertheless much more convoluted.

The constraint imposed on each $\langle a, b, k\rangle \in R(\langle u, v, l\rangle)$ by Operator 4 is generated by compacting twice, each time for either $\langle u, v, l\rangle$ or $\langle a, b, k\rangle$:

- For $\langle a, b, k\rangle$, the compacted set $\boldsymbol{A}$ is a subset of $\lambda(b)$. Each $e'' \in \boldsymbol{A}$ $(e'' \neq \langle a, b, k\rangle)$ eventually "falls" into $\lambda(b)$ by "walking" along a path that traverses $\langle a, b, k\rangle$, i.e., $R(e'')$ contains a ρ-path traversing $\langle a, b, k\rangle$. We can imagine $\boldsymbol{A}$ as a "gourd" hanging under the "handle" $\langle a, b, k\rangle$, as depicted in Figure 4(a).
- For $\langle u, v, l\rangle$, the set $\boldsymbol{B}$ is compacted from the set $\boldsymbol{C} = \{\langle c, d, j\rangle \in \boldsymbol{A} | \{\langle u, v, l\rangle, \langle a, b, k\rangle\} \subseteq [R(\langle c, d, j\rangle) \cap \boldsymbol{A}]^{b}_{d}\} \subseteq \boldsymbol{A}$. Regarding $\langle u, v, l\rangle$ as a "handle", then obviously $\boldsymbol{B} = \chi^{u}_{R(E)}(C) \subseteq \boldsymbol{A}$ and $\boldsymbol{B}$ is also like a "gourd" hanging under the "handle" $\langle u, v, l\rangle$, as depicted in Figure 4(b).
- Intuitively speaking, if $\langle a, b, k\rangle$ is kept in $R(\langle u, v, l\rangle)$, there must exist $P = S - \cdots - u \subseteq E$ such that $\{\langle u, v, l\rangle, \langle a, b, k\rangle\} \subseteq R(e''')$ for each $e''' \in P$. Meanwhile, all those paths like P must fulfill the strict constraint that: suppose all the edges on those paths form a set ES, then $\chi^{u}_{R(E)}(ES) \neq \emptyset$.

[4] $\emptyset$,{} are not distinguished in the paper. The result of set operations can be united with {} to avoid null reference.

It should also be noted that, the result of Operator 4 does not depend on the order of edge choice during the iteration (see Theorem 3 in Section 7).

The sets $\boldsymbol{A}$ and $\boldsymbol{B}$ can be computed within $|E| * O(|E|^3)$ and therefore step (1) can be finished in $|E| * |E| * O(|E|^3)$. The execution will terminate before it reaches $|E|$ iterations, since at least one edge is pruned per round. Overall, the cost is $|E| * |E| * |E| * O(|E|^3) = O(|E|^6)$.

3.2 The ZH algorithm, the temporal cost and the necessity proof

With the above basic operators, the ZH algorithm can be henceforth given in the following Algorithm 1. Detailed motivation of the algorithm will be discussed in Section 4.

Algorithm 1: the ZH algorithm

Input: $G =< V, E, S, D, L, \lambda >$ in $2-\mathrm{MSP}$

Output: 'yes' or 'no' decision on σ-path existence

1. $R(E) \leftarrow \{R(e)|R(e) \leftarrow \rho(e), e \in E\}$
2. **for** $e = \langle u, v, l\rangle \in E$ $(2 \leq l < L)$
   call $\psi_{R(E)-\{R(e)\}}(R(e))$ to prune $R(e) \in R(E)$
3. **repeat** step 2 **until** each $R(e) \in R(E)$ becomes stable
4. G contains a σ-path iff. $\chi^{D}_{R(E)}(\lambda(D)) \neq \emptyset$

Initially in step 1, the edges contained in $R(e)$ are as defined by Operator 2, and let it be denoted by

$$R_0(e) =_{\mathrm{def}} \rho(e),\ R_0(E) =_{\mathrm{def}} \{R_0(e)|e \in E\}. \tag{1}$$

$R(e)$ is pruned thereafter with $|R(e)| \leq |E|$ decreasing monotonically, until this procedure eventually stops.

In step 2, Operator 4 utilizes $(R(E) - \{R(e)\})$ to restrict each $e' \in R(e)$ for the determination of σ-path existence, by not only detecting the "future" (using the ρ-paths in $R(e)$) but also summarizing the "history" (by binding $(R(E) - \{R(e)\})$ with $R(e)$).

To help better grasp the detailed working mechanism of the ZH algorithm, some motivating running instances (on $\mathrm{K-SAT}$) are provided in Section 14. As also discussed in that section, the algorithm has been validated on a wide range of test cases, including a large number of hard $3-\mathrm{SAT}$ instances of moderate sizes generated by a phase-transition-theory based model [XL00, XBH+07]. Refer to our online tutorials for supplementary video demos of the algorithm.

Definition 5 (The compact kernel). The resulted $\chi^{D}_{R(E)}(\lambda(D))$ in step 4 of the ZH algorithm is called *the compact kernel of* G.

Our result is amazingly simple, given as the following conjecture:

Conjecture 1 (The Compact Kernel Conjecture). *G contains a σ-path iff. the compact kernel of G is not empty.*

Theorem 4 (The cost). *The cost of the ZH algorithm can be* $O(|E|^9)$.[5] (Proved in Section 8)

Theorem 5 (The necessity). *If* G *contains a* σ-path*, then the compact kernel of* G *is not empty.* (the direction of necessity is naturally trivial; proved in Section 9)

3.3 The sufficiency proof

Before the proof of sufficiency, two notations, one metric and a specially constructed algorithm need to be defined.

Definition 6 (*ES*[*i*: *j*]). Let $ES \subseteq E$. $ES[i\colon j]$ denotes the set of all edges of stages from i to j in ES, where $1 \leq i \leq j \leq L$. If $i > j$, $ES[i\colon j] = \emptyset$.

Definition 7 (*ZH* \ *step*4). $ZH \setminus step4$ stands for all the steps of the ZH algorithm except step 4.

To apply mathematical induction, the following metric for G is required.

Metric 1. $f(G) = \sum_{v \in V - \{S,D\}} (d^-(v) - 1)$.

The ZH algorithm is then embedded in a **Proving algorithm** (abbr. PA, see Algorithm 2), which is specially constructed to set up the sufficiency proof.

Algorithm 2: the Proving algorithm

Input: $G =< V, E, S, D, L, \lambda >$ in $2-$MSP

Output: 'yes' or 'no' decision on σ-path existence in step 4 of the PA

1. apply $ZH \setminus step4$ on G to generate $R_0(E)$ and the stable $R(E)$
2. $ES1 \leftarrow \chi^D_{R(E)}(\lambda(D))$
3. **if** $[ES1_{sub}]^D_S[L\colon L] = ES2$ (where $ES1_{sub}$ should obey the criteria (i),(ii),(iii),(iv),(v))
   then $(\forall \langle w, D, L\rangle \in [ES1_{sub}]^D_S)(\exists\ \sigma - \text{path}\ S - \cdots - w - D \subseteq ES1_{sub})$

   The criteria on the constitution of $ES1_{sub}$ are as follows:

   (**i**) $ES1_{sub} = ES2 \cup \left(\left(\bigcup_{\langle y,D,L\rangle \in ES2} \lambda_{sub}(y) \right) \cap ES1 \right)$, where $\emptyset \neq ES2 \subseteq ES1[L\colon L]$.

   (**ii**) $\lambda_{sub}(y) \subseteq \lambda(y)[1\colon 1] \cup \{e \in \lambda(y)[2\colon L] | \langle y, D, L\rangle \in R(e)\}$, where $\langle y, D, L\rangle \in E$.

   (**iii**) If a σ-path $P = S - \cdots - y - D \subseteq ES1_{sub}$, then $P[1\colon L-1] \subseteq \lambda_{sub}(y)$.

   (**iv**) $ES1_{sub}$ contains one of $\langle *, v, k\rangle$ at most for each $v \in V_k$ $(1 < k < L)$.

   (**v**) If $|ES2| > 1$, there exists $S - a_1 - \cdots - a_i \subseteq [ES1_{sub}]^D_S$ $(i > 1)$ such that $[ES1_{sub}]^D_S$ contains one $\langle a_j, *, j+1\rangle$ at most for each a_j $(1 \leq j < i)$ while two $\langle a_i, *, i+1\rangle$ at least, and $S - a_1 - \cdots - a_i \not\subseteq \lambda_{sub}(y)$ for $\langle y, D, L\rangle \in ES2$.

Step 1 of the PA is actually the ZH algorithm. The PA only makes a sufficient judgment.

The σ-path as claimed by step 3 (called the *solution* found by the PA) should be a subset of $ES1_{sub} \subseteq ES1$, fulfilling five criteria:

- The composition of $ES1_{sub}$ is defined by criterion (i), where the subset $ES2 \subseteq ES1[L\colon L]$ is used to control the last stage; and then for each $\langle y, D, L\rangle \in ES2$, the subset $\lambda_{sub}(y) \subseteq$

[5] |A| is used to denote the cardinality of the set A.

$\lambda(y)$ is used to constitute the rest of $ES1_{sub}$.

- The definition of $\lambda_{sub}(y)$ by criterion (ii) is based on a key insight that each pair of edges "$\langle y, D, L\rangle, e$" involved in the computed relation "$\langle y, D, L\rangle \in R(e)$" by the ZH algorithm can be exploited to find the solutions of the PA.
- Criterion (iii) defines the relation between $\lambda_{sub}(y)$ and $ES1_{sub}$. Since $P = S - \cdots - y - D$ is a σ-path, $P[1\colon L-1] \subseteq \lambda(y)[1\colon 1] \cup \{e \in \lambda(y)[2\colon L] | \langle y, D, L\rangle \in R(e)\}$. Generally, $\lambda_{sub}(y) \subseteq \lambda(y)[1\colon 1] \cup \{e \in \lambda(y)[2\colon L] | \langle y, D, L\rangle \in R(e)\}$. If $P \subseteq ES1_{sub}$, $\lambda_{sub}(y)$ is demanded to contain the whole $P[1\colon L-1]$.
- By criterion (iv), $[ES1_{sub}]_S^D[1\colon L-1]$ should be a tree.
- Criterion (v) defines the relation between different $\lambda_{sub}(y)$.

The connectivity prerequisite $[ES1_{sub}]_S^D[L\colon L] = ES2$ and the additional requirement for the demanded σ-path as a solution put further limits on the content of $\lambda_{sub}(y)$, thereby providing strong maintanence of mathematical equivalence during our proposed inductive proof of sufficiency.

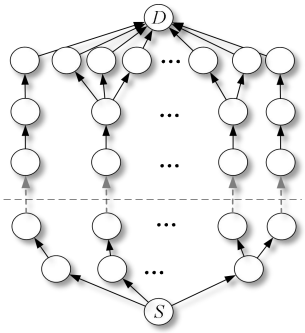


Figure 5: Illustration of Lemma 1

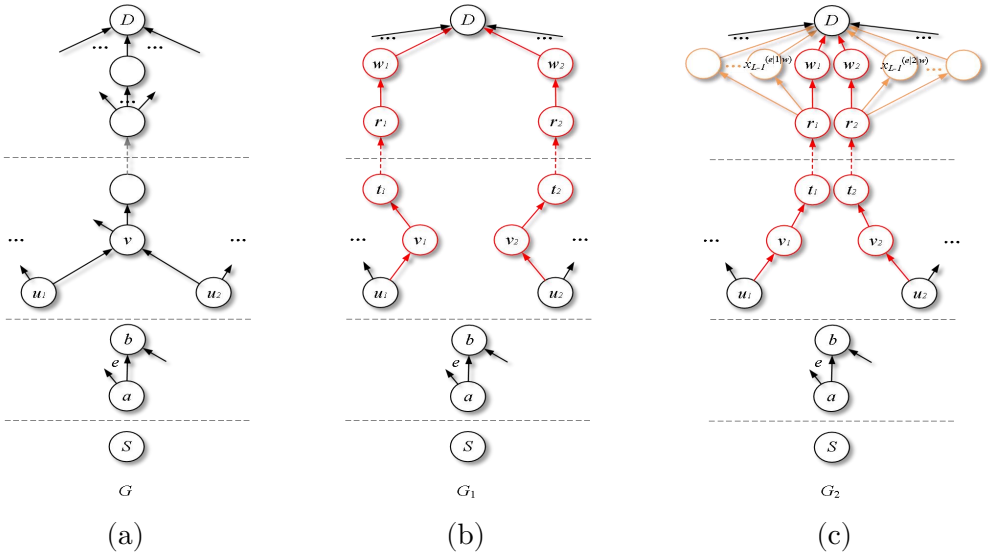


Figure 6: Illustration of Lemma 2 ($v \in V_l, 1 < l < L$; recall $L \geq 5$ by Definition 1)

The proving framework of mathematical induction on $f(G)$. By Definition 3(b) (item 1), we have $f(G) \geq 0$ when applying Metric 1. For any G, if $f(G) = 0$, it can be proved that the PA

can make a correct decision (see the following Lemma 1). Assuming that the PA can make a correct decision for any G' that $f(G') < m$ $(m > 0)$ (**H1**), we can prove that the PA can make a correct decision for any G that $f(G) = m$ (see the following Lemma 2).

The major difficulty and challenge of the above mathematical induction-based proof is that, given the set $ES1_{sub}$ for the input G, we shall construct a mathematically equivalent new $ES1_{sub}'$ for some other graph G' that is "smaller" than G. Indeed, some parallels can be drawn (as will be revealed during the proof) on the logical power of construction between the $ES1_{sub}'$ by our approach and the uncomputatble functions [Chu36, Tur36] by diagonalization.

Lemma 1. *Let $G =< V, E, S, D, L, \lambda >$ be the input to the PA and there is no multi-in-degree vertex at stage 1,2,..., $L-1$ in G (see Figure 5). After applying the PA on G, if $[ES1_{sub}]_S^D[L\colon L] = ES2$, then $(\forall\langle w, D, L\rangle \in [ES1_{sub}]_S^D)(\exists\ \sigma - \text{path}\ S - \cdots - w - D \subseteq ES1_{sub})$.*

Lemma 2. *Given the mathematical induction hypothesis H1 that the PA can make a correct decision for any G' that $f(G') < m$ $(m > 0)$. Let $G =< V, E, S, D, L, \lambda >$ be the input to the PA, $f(G) = m$, the vertex v of stage l $(1 < l < L)$ be a multi-in-degree vertex, and there exists no multi-in-degree vertex (except D) above stage l (see Figure 6(a)). After applying the PA on G, if $[ES1_{sub}]_S^D[L\colon L] = ES2$, then $(\forall\langle w, D, L\rangle \in [ES1_{sub}]_S^D)(\exists\ \sigma - \text{path}\ S - \cdots - w - D \subseteq ES1_{sub})$.* (the major hard stone; sketched in Section 11.1 and proved in Section 11.2-11.3)

The αβ lemma (Summarizing Lemma 1,2). *Let $G =< V, E, S, D, L, \lambda >$ be the input to the PA. After applying the PA on G, if $[ES1_{sub}]_S^D[L\colon L] = ES2$, then $(\forall\langle w, D, L\rangle \in [ES1_{sub}]_S^D)(\exists\ \sigma - \text{path}\ S - \cdots - w - D \subseteq ES1_{sub})$.*

Theorem 6 (The sufficiency). *If the compact kernel of G is not empty, there exists a σ-path in G.* (proved in Section 12, using the αβ lemma)

Combining Theorem 1,4,5,6, we can eventually prove Conjecture 1 and obtain the following theorem.

Theorem 7 ($\mathbf{NP = P}$). *There exists a poly-time algorithm for $2-MSP$, i.e., there exists a poly-time algorithm for NP-complete problems.*

4 The insights

While efforts on the P vs. NP problem are diversified [Mel07, Raz85, Wil14, MW18, AW08, Hak85, MS01, BI11, Mul12, Blu17, DDX+23, XG23], there has been little progress and existing approaches are either broken, stalled, or being far way from the ultimate proof [TJR75, Raz89, RR97, Aar18, Tar87, BIP16, IP16, GIM+20]. For example, one of the mostly widely followed approach in recent years is the program of *geometric complexity theory* (GCT) [MS01, BI11, Mul12, BIP16, DIP19, IK20, BDI21, DDS21], which appears to avoid the well-known barriers of other methods, offers the rich mathematical tools in algebraic geometry, and has achieved a number of proof-of-concepts (e.g., some modest lower bounds on matrix multiplication has been presented by [BI11]). Nevertheless, now the method begins to receive some doubts on its feasibility, especially with recent disproofs [BIP16, IP16, GIM+20] of its several important conjectures. What's worse, the techniques developed by such approaches can be barely utilized for the other direction (i.e., proving $\mathrm{P = NP}$); and simple mildest improvements to known

algorithms [Woe03, FK13] are just insufficient to break the large complexity barrier. Therefore, we resort to developing our own techniques from scratch[6]—namely, (1) the MSP problem and the proving framework of mathematical induction on the metric $f(G)$ and (2) the ZH algorithm.

The insights of our approach are summarized as follows.

4.1 On the MSP problem structure

Unlike other well-known NP-complete problems, the MSP problem is a crafted "unnatural" problem. It is a common practice to concentrate a study on a more convenient novel problem than the original well-known ones—for one example, the focus of the GCT program is a family of orbit closure and stabilizer problems reduced from the lower bound problem arising in the context of the P vs. NP question; for another example, the quasipoly-time lower bound for Graph Isomorphism was obtained when directly solving another poly-time equivalent problem (under Karp reductions), i.e. String Isomorphism. The major bonus brought by the MSP problem structure is two-fold, as follows.

4.1.1 The linear-order metric $f(G)$ and the inductive proving framework

We have been working on the MSP problem for such a long time, because we have been intrigued by a metrizable graph property with respect to the structure of MSP. It is believed to be the key towards the design of efficient exact algorithms for the problem.

All MSP instances can be arranged in a sequence according to the quantitative linear-order metric $f(G) = \sum_{v \in V - \{S,D\}} (d^-(v) - 1)$ (see Metric 1). The problem structure of MSP makes us easier to construct mathematical and algorithmic equivalent instances in the above linear-order sequence, for the inductive proof of the correctness of the algorithm.

Given an arbitrary instance I_{cur} in the sequence. Suppose $d^-(v) > 1$ for some $v \notin \{S, D\}$ in I_{cur}, as shown in Figure 6(a). We can construct an instance I_{pre}, such that I_{cur} and I_{pre} keep some sense of mathematically equivalence on the target property concerned by us. The convenience of such a construction originates from the problem structure of MSP: what the construction need to do, is just following the structure and labels of I_{pre} and defining a different but essentially equivalent set of labels for I_{cur}.

This makes it become our persistence to find an algorithm that can fulfill the above proving framework of mathematical induction based on mathematical equivalence. Until the ZH algorithm appears to our mind.

4.1.2 The system invariant $ES1_{sub}$ and the conservative expansion

A crucial discovery is made on a system invariant (i.e., the $ES1_{sub}$ in the PA) between mathematical equivalent MSP instances. This system invariant is used in combination with a "*conservative expansion*" technique, which will be described as follows.

During the inductive proof of the correctness of the proposed ZH algorithm, the algorithm itself is actually used as a "reasoning system". Hence, what we only need to do is just to guarantee that the computed results (indeed they are sets of edges) by the actions of ZH algorithm on MSP instances of different order (i.e., the I_{cur} and I_{pre} measured by $f(G)$) can keep essentially the same.

---

[6] This is a bit similar to the research story of the AKS algorithm [AKS04] for Primality Test. It had been quite shocking on the originality and simplicity of the AKS prime test, given that previous researchers had made much more complicated and modern efforts on theories and methods to attack the problem (often involving great ingenuity); the success was supposed to be contributed to the clever and original combination of classical ideas [Gra04].

To provide such guarantee for the "reasoning" of the ZH algorithm, we firstly radically expand the labels of I_{pre} (as shown in Figure 6(b)). That is, the labels are expanded to include as many edges as possible. In this way, it is much easier for the confirmation of the computed result by the ZH algorithm on I_{pre}. That's because, according to the reachability of an edge e (i.e., the $R(e)$ defined in the paper), "larger" labels can give e more chances to "go through" the paths in $R(e)$. This is just as told by the anecdotes of *Isaac Newton's Door with Two Cat Holes*—the little kittens could definitely follow their mother through the larger hole, as long as they can pass through the smaller one. We can hence easily infer the existence of σ-paths (potential solutions) in I_{pre}, by the proposed framework of mathematical induction on $f(G)$.

While the radical expansion provides such convenience, it might potentially bring in extra solutions for I_{pre} when compared with I_{cur} and hence make the two instance become less equivalent. Thus, some method of control is needed to ensure that no more solutions which we care about can be introduced, hence making the radical expansion actually become conservative. The aforementioned system invariant $ES1_{sub}$ serves for this purpose.

The existence of $ES1_{sub}$ has a similar logical power to the existence of uncomputable functions [Chu36, Tur36]: (1) initially, we use the logical power endowed by the inductive hypothesis to strictly "squeeze out" each such above potential σ-paths—just an analogy of a function "$f_\alpha(x)$" computed by a Turning machine (represented by the string α and with the input x); (2) then, we precisely list out the σ-paths one by one (as shown in Figure 6(c))—just an analogy of the sequence of all computable functions; (3) finally, we find the system invariant $ES1_{sub}$ for I_{pre} guided by the $ES1_{sub}$ for I_{cur}, and further determine the solutions actually demanded by the algorithm through logic inference—just an analogy of the inference of the uncomputable function "$f_x(x)+1$"; (4) subsequently, the existence of global solutions in I_{cur} can be henceforth constructed.

4.2 On the tackling of the complexity

To tackle the hardness, Lance Fortnow [For09, For21] categorized some of the tools one can use on NP-complete problems, i.e., brute force [ABC+98], parameterized complexity [DF12], approximation [Aro98,GW95] and heuristics & average-case complexity [Lev86,SAT23]. Most exact algorithms for NP-complete problems (similar for NP-hard problems) in the literature involve either dynamic programming across the subsets, pruning the search tree, preprocessing the data, or local search [Woe03, FK13]. Though significant progress (including but not limited to [Bjö14, BHK09, Wil05]) in the area of exhaustive search has been made in recent decades, existing methods still failed on the formidable exponential barrier. The incapability of those methods mainly lies in that: once losing the help of exhaustive enumeration, the methods just failed to continue to accurately identify the information needed to make the correct global decision. The key to our overcoming of this barrier is two-fold, as follows.

4.2.1 The edge-set representation of paths

When dealing with paths, traditional exact graph algorithms usually need to explicitly represent each of them as an independent path. Instead, our method treat paths from an edge-set viewpoint, i.e., they are represented by a set of edges traversed by them. Thus, the cost is reduced to polynomial time.

Nevertheless, the representation of paths based on edge sets inevitably arouses ambiguity—a non-empty edge set can be determined by a path, while it may not work vice versa. An algorithm designed to satisfy our proposed proving framework of mathematical induction on

$f(G)$, as described above, provides us a chance to logically prove that: a computed non-empty set of edges by a series of strong constraints (e.g., the compact kernel in the ZH algorithm) can determine the existence of a path with global property (e.g., the demanded σ-path).

4.2.2 The computation of the reachable-path edge-set $R(e)$, and the discovery of the relation between local and global strategies

A novel mechanism of the interplay between local strategies and global strategies is discovered and established.

A computational property named the reachability of an edge (i.e., $R(e)$, see Operator 2) is defined and adopted, which can be utilized to summarize the "history" and to detect the "future" for searching "global paths". Contemporarily, the reachability of one edge is forced to be constrained by the reachability of the other ones (see Operator 4 and the ZH algorithm). This rightly establishes a recursive relation of the reachability of edges of different stages in the multi-stage graph.

The recursive relation we exploited resembles the state-transition equation in dynamic programming—a standard approach for getting fast exact algorithms for NP-complete problems [Woe03, HK62, HS78, Law76, LLR+85, Epp01], while the former one appears to be much more convoluted. Nevertheless, since all computations involved can decrease monotonically, such type of algorithm is destined to be poly-time upper-bounded.

The proof based on our proposed proving framework of mathematical induction on $f(G)$ (as discussed in Section 4.1) provides a firm guarantee for the established recursive relation. The design of the basic operators and the adjustment of $2-\mathrm{MSP}$ from MSP are also performed largely through logical reasoning to support the proposed proving framework rather than through mere intuition. This is similar to the studies of Ramanujan Summation of "$1+2+3+4+\cdots=-\frac{1}{12}\,(\mathfrak{R})$" [Ram14], Gödel Incompleteness Theorem [Göd31], uncomputable functions [Chu36, Tur36], etc., where the motivations and insights were characterized by logical reasoning instead of misleading experiential intuition.

As an aside, it is worth noting that we have tried to re-write the proofs of several long-existing algorithms using our proposed inductive proving framework. Though we did not discover any brand-new algorithm of better performance, our proving framework did help to find and prove algorithms. For instance, in the case of the well-known Single-Source Shortest Path (SSSP) problem for multi-stage graphs, the correctness of the classic dynamic programming algorithm can be quickly and fluidly verified by mathematical induction on the linear-order metric $f(G)$.

5 Proof of Theorem 1

Theorem 1. $2-\mathrm{MSP}\in\mathrm{NPC}$.

Proof. A number of NP-complete problems can been polynomially reduced to the MSP problem (see [JLW+14, FJP14]).

Literature [JLW+14] gave a reduction from $\mathrm{CNF}-\mathrm{SAT}$. Assume each CNF consists of at least 4 clauses. The reduction is performed as follows:

(i) Each literal of a clause in a $\mathrm{CNF}-\mathrm{SAT}$ instance corresponds to a vertex of $G=\langle V,E,S,D,L,\lambda\rangle$ in a MSP instance.

(ii) Each clause in the $\mathrm{CNF}-\mathrm{SAT}$ instance becomes one stage of vertices in G.

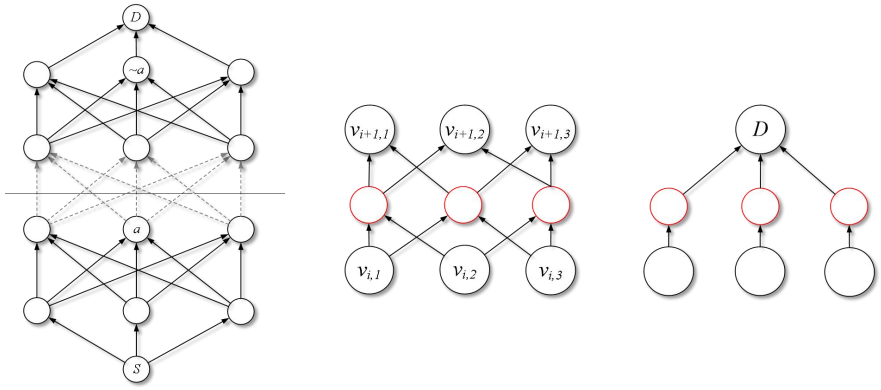
(iii) Two vertices S and D are added to G.

(iv) All vertices between adjacent stages are fully connected.

(v) The labels are defined as follows: set $\lambda(D)=E$; for each vertex $x\in V-\{S,D\}$ (assume x corresponds to some literal p), set

$$\lambda(x)=E-\left\{e\middle|\begin{array}{l}e \text{ starts from } \overline{x} \text{ or end at } \overline{x}, \text{ where the vertex } \overline{x}\\ \text{corresponds to the complementary literal of } p\end{array}\right\}. \tag{2}$$

An example of the reduction from $3-\mathrm{SAT}$ to MSP is shown in Figure 7(a). In this sense, MSP establishes a graph model for $\mathrm{CNF}-\mathrm{SAT}$.



(a) (b) (c)

Figure 7: Reduction from 3-SAT to 2-MSP

To further reduce $3-\mathrm{SAT}$ to $2-\mathrm{MSP}$, we just need to replace edges between each pair of adjacent stages (except the first two and last two stages) with a stage gadget as shown in Figure 7(b). The three vertices at the lower stage (e.g., $v_{i,1}, v_{i,2}, v_{i,3}$ in Figure 7(b)) are organized as the combination of "C_3^2" to "enter" the three auxiliary vertices in the gadget; and then the auxiliary vertices "enter" the vertices at the upper stage (e.g., $v_{i+1,1}, v_{i+1,2}, v_{i+1,3}$ in Figure 7(b)) in the same mode.

In addition, replace the edges between the last two stages with a stage gadget as shown in Figure 7(c). An auxiliary vertex is inserted between each vertex at the lower stage and the vertex D.

The labels of the auxiliary vertices can all be set to the updated E. The labels of the original vertices should be recomputed (by the above step (v) of the reduction from $\mathrm{CNF}-\mathrm{SAT}$ to MSP) to follow the change of E.

A complete view of the reduction is illustrated in Figure 1(c). See Section 14 for a more concrete example of the reduction.

It takes little effort to exam just item-by-item, that the resulted instance fulfills Definition 3(b). Note that, in the case of $2-\mathrm{MSP}$, we can relax our assumption such that each CNF consists of at least 2 clauses.

The $3-\mathrm{SAT}$ problem is therefore poly-time Karp-reducible [Kar72] to the $2-\mathrm{MSP}$ problem.

For one direction, it's easy to see by Definition 2 that, the $3-\mathrm{SAT}$ instance must can be satisfied, if there exists some σ-path in the corresponding $2-\mathrm{MSP}$ instance.

In fact, if $v_0-v_1-\cdots-v_2-\cdots-v_L\subseteq E$ (where $v_0=S$, $v_L=D$) is a σ-path, then each

v_{2*i-1} ($i \in \{1,2,\ldots,N\}$, $N \geq 2$ is the number of clauses) on the path must be a vertex that stands for a literal (let it denoted by $p_{v_{2*i-1}}$) in a clause C_i of the $3-$SAT problem, according to the above reduction.

Since each $\lambda(v_{2*i-1})$ ($i \in \{1,2,\ldots,N\}$) excludes edges that start from or end at $\overline{x}$ (where $\overline{x}$ corresponds to the complementary literal of $p_{v_{2*i-1}}$), no pair of literals among $\left\{p_{v_{2*j-1}} \middle| j \in \{1,2,\ldots,N\}\right\}$ is complementary. We thus know that there must exist an assignment φ that satisfies all these $p_{v_{2*j-1}}$ ($j \in \{1,2,\ldots,N\}$). This assignment φ also satisfies the given $3-$SAT problem.

For the other direction, if the $3-$SAT instance is satisfied by some assignment φ, then there must exist some literal p_i ($i \in \{1,2,\ldots,N\}$, $N \geq 2$ is the number of clauses) in each clause C_i such that $\varphi(p_i) = \text{true}$.

Then, by the above definition of the labels and by Definition 2, the path $S - x(p_1) - x_1 - x(p_2) - x_2 - \cdots - x(p_N) - x_N - D$ (where $x(p_i)$ stands for the vertex created corresponding to p_i, x_i stands for an inserted auxiliary vertex, $i \in \{1,2,\ldots,N\}$) in the corresponding $2-$MSP instance must be a σ-path, according to the reduction.

Therefore, $2-$MSP is NP-complete.[7] □

6 Observations of the basic operators

In the context of the ZH algorithm, where $R(E) = \left\{ R(e) \middle| \begin{matrix} R(e) = [R(e)]_v^D \subseteq R_0(e), \\ e = \langle u, v, l \rangle \in E \end{matrix} \right\}$, the following observations can be easily justified. They can serve as exercises for the understanding of the basic operators and the proving details of the correctness of the ZH algorithm. Feel free to step over this part if the operators are already well grasped.

Observation 1 (χ, subset).
(1) Obviously, for arbitrary $v \in V - \{S\}$ and arbitrary $ES \subseteq E$, $\chi^v_{R(E)}(ES) = \chi^v_{R(E)}([ES]^v_S) = \chi^v_{R(E)}\left(\chi^v_{R(E)}(ES)\right) \subseteq ES$.
(2) Suppose $\lambda(D) = E$. For arbitrary $v \in V - \{S\}$ and arbitrary $ES \subseteq E$, we have $\chi^v_{R(E)}(ES) \subseteq \lambda(v)$, since each $e \in \chi^v_{R(E)}(ES)$ is required to traverse v by some ρ-path in $R(e) \subseteq R_0(e)$.

Observation 2 (χ, single-in-degree vertex). Suppose $\lambda(D) = E$. If $P = u_i - \cdots - u_j \subseteq \chi^{u_j}_{R(E)}(ES)$ (where indices indicate the stages of vertices here), u_h is a single-in-degree vertex for each $i < h < j$ and $ES[j{:}\,j]$[8] $= P[j{:}\,j]$, then P is a ω-path.

Proof. Let's show $\langle u_{h-1}, u_h, h \rangle \in \bigcap_{h \leq m \leq j} \lambda(u_m)$ for $i < h < j$. By the definition of Operator 3 for $\chi^{u_j}_{R(E)}(ES)$, $\left[R(\langle u_{h-1}, u_h, h \rangle) \cap \chi^{u_j}_{R(E)}(ES)\right]^{u_j}_{u_h} \neq \emptyset$ and the edges in $\left[R(\langle u_{h-1}, u_h, h \rangle) \cap \chi^{u_j}_{R(E)}(ES)\right]^{u_j}_{u_h}$ must be on the unique path $u_h - \cdots - u_j$. This is because, there exists only one path from u_h to u_j in ES, as $u_h, u_{h+1}, \ldots, u_{j-1}$ are single-in-degree vertices and $ES[j{:}\,j] =$

[7] Indeed, its NP-completeness is so trivially verifiable even for non-specialist readers, that we have published more than ten proofs and we even assigned the proof as a small homework to hundreds of graduate students in an algorithms & complexity course for many consecutive years. In several seminars when visiting other universities, tens of students proposed at least 6 independent approaches of reduction, as we know.
[8] See Definition 6.

$P[j\colon j]=\{\langle u_{j-1},u_j,j\rangle\}$. Then, we have $u_h-\cdots-u_j\subseteq R(\langle u_{h-1},u_h,h\rangle)\subseteq R_0(\langle u_{h-1},u_h,h\rangle)$. By the definition of Operator 2, $\langle u_{h-1},u_h,h\rangle\in\bigcap_{h\le m\le j}\lambda(u_m)$.

If $u_j\neq D$, then by the definition of Operator 3 for $\chi^{u_j}_{R(E)}(ES)$ and due to $\langle u_{j-1},u_j,j\rangle\in\chi^{u_j}_{R(E)}(ES)$, we have $\langle u_{j-1},u_j,j\rangle\in\lambda(u_j)$. Otherwise, if $u_j=D$, by the assumption that $\lambda(D)=E$, we also have $\langle u_{j-1},u_j,j\rangle\in\lambda(u_j)=\lambda(D)$.

Therefore, P is a ω-path by Definition 2. □

Observation 3 (ψ, equivalence). By the definitions of Operator 2,3, for the designated $e=\langle u,v,l\rangle$ and $e'=\langle a,b,k\rangle$ in the definition of Operator 4, we can straightforwardly obtain

$$\begin{aligned}\boldsymbol{A}&=\chi^{b}_{R(E)}(\{\langle x,y,i\rangle\in E\big|e'\in[R(\langle x,y,i\rangle)\cap\lambda(b)]^b_y\}\cup\{e'\})\\&=\chi^{b}_{R(E)}\left(\left[\{e'\}\cup\left\{e''\in E\middle|\begin{array}{c}e''=\langle x,y,i\rangle\ (i<k),\text{ where}\\[R(e'')\cap\lambda(b)]^b_y\text{ contains a path}\\ y-\cdots-b\text{ that traverses }e'\end{array}\right\}\right]^{b}_{S}\right)\end{aligned}\tag{3}$$

and

$$\begin{aligned}\boldsymbol{B}&=\chi^{u}_{R(E)}(\{\langle c,d,j\rangle\in\boldsymbol{A}\big|\{e,e'\}\subseteq[R(\langle c,d,j\rangle)\cap\boldsymbol{A}]^b_d\})\\&=\chi^{u}_{R(E)}\left(\left[\left\{e'''\in E\middle|\begin{array}{c}e'''=\langle c,d,j\rangle\in\boldsymbol{A}\ (j<l),\text{ where}\\[R(\langle c,d,j\rangle)\cap\boldsymbol{A}]^b_d\text{ contains a path }d-\cdots-b\\ \text{that traverses both }\langle u,v,l\rangle\text{ and }\langle a,b,k\rangle\end{array}\right\}\right]^{u}_{S}\right).\end{aligned}\tag{4}$$

Observation 4 (ψ, set $\boldsymbol{A}$,$\boldsymbol{B}$,$\boldsymbol{C}$). $B\subseteq C\subseteq A$.

This is straightforward, see Observation 1.

Observation 5 (ZH algorithm, subset). After the computation of the ZH algorithm:

(1) Obviously, $P\subseteq\chi^{D}_{R(E)}(D)$ if $P\subseteq E$ is a σ-path.

(2) If $P=a_0-\cdots-a_{i-1}-a_i-\cdots-a_L\subseteq E\quad(\ 0<i<L\ )$ is a σ-path, then $P[i+1\colon L]\subseteq R(\langle a_{i-1},a_i,i\rangle)$.

(3) Suppose we expand some lables of G and recompute $R(E)$ using the ZH algorithm. Let the new set be denoted by $R'(E)$. Then, $P\subseteq R(e)$ would imply $P\subseteq R'(e)$ (where P is a path in G, $e\in E$, $R'(e)\in R'(E)$).

(4) For arbitrary $e\in E$ and arbitrary $ES\subseteq E$, $ES\subseteq R_0(e)$ if $ES\subseteq R(e)$.

Observation 6 (ZH algorithm, after split & expansion). Let $G=<V,E,S,D,L,\lambda>$, where $d^-(v)=2$ for some $v\in V_l$ $(1<l<L)$ and $d^-(\bar v)\le 1$ for arbitrary $\bar v\in V_j$ $(l<j<L)$ (analogous to Figure 6(a)). Let $G'=<V',E',S,D,L,\lambda'>$ be a graph constructed from G (analogous to Figure 6(b)), where:

(i) $V'=(V-\{v|v\text{ is on }v-\cdots-D\subseteq E\})\cup\bigcup_{i\in\{1,2\}}\{\hat x_i^l,\cdots,\hat x_i^{L-1}\}\cup\{D\}$;

(ii) $E' = \left(E - \left\{e \middle| \begin{matrix} e \in u_i - v - \cdots - D \subseteq E, \\ i \in \{1,2\} \end{matrix}\right\}\right) \cup \left\{e \middle| \begin{matrix} e \in u_i - \hat{x}_i^l - \cdots - \hat{x}_i^{L-1} - D, \\ i \in \{1,2\} \end{matrix}\right\}$;

(iii) $(\lambda(x) \text{ of } G')$[9] $= (\lambda(x) \text{ of } G)$ $(x \in (V' \cap V - \{D\}))$;

(iv) $(\lambda(y) \text{ of } G') = E'$ $(y \in V' - V)$ and $(\lambda(D) \text{ of } G') = E'$.

After applying the ZH algorithm on G and G', we have

$$\begin{gathered} \left(\chi^D_{R(E)}(\lambda(D)) \text{ of } G'\right) \supseteq \\ \left(\begin{matrix} \left(\left(\chi^D_{R(E)}(\lambda(D)) \text{ of } G\right) - \left\{e \in P = u_i - v - \cdots\cdots - D \middle| P \subseteq E, i \in \{1,2\}\right\}\right) \\ \cup \left\{e \in P = u_i - \hat{x}_i^l - \cdots - \hat{x}_i^{L-1} - D \middle| \begin{matrix} (R(\langle u_i, v, l\rangle) \text{ of } G) \neq \emptyset, \\ P \subseteq E', i \in \{1,2\} \end{matrix}\right\} \end{matrix}\right). \end{gathered} \tag{5}$$
[10]

Proof. The reason is that, all labels of newly introduced vertices in G', after the "split" of the vertex v in G, are set to be E'. Hence, it is straightforward (by plain examination of the definitions of Operator 2,3,4 for G and G') to see that, the existence of a path from s to D in $\left(\left[R(\langle r, s, k\rangle) \cap \chi^D_{R(E)}(\lambda(D))\right]^D_s \text{ of } G\right)$ $(\langle r, s, k\rangle \in E, 1 \leq k < L)$ implies the existence of another path from $\hat{s}$ to D in $\left(\left[R(\langle \hat{r}, \hat{s}, k\rangle) \cap \chi^D_{R(E)}(\lambda(D))\right]^D_{\hat{s}} \text{ of } G'\right)$ $(\langle \hat{r}, \hat{s}, k\rangle \in E')$. To conclude this observation, we shall further note that $(\lambda(D) \text{ of } G') = E'$. □

7 The theorems of equivalence & uniqueness for the basic operators

We now rewrite Operator 3 and 4 in "analytic forms".

Operator 3 ($\chi^v_{R(E)}(ES)$, analytic form). Given $ES \subseteq E$, $v \in V_l$ and the collection of ρ-path edge-sets $R(E)$. Give Operator 3 as the "*analytic form*": $\chi^v_{R(E)}(ES) =_{\text{def}} ES1$, s.t. $ES1 \subseteq ES$, $\mathfrak{X}(ES1) = true$ and $\mathfrak{X}(ES2) = false$ $(ES2 \subseteq ES)$ for each possible $ES2 \supset ES1$, where

$$\mathfrak{X}(\Delta) =_{\text{def}} \left(\Delta = \left[\left\{e = \langle a, b, k\rangle \in \Delta \middle| \begin{matrix} [R(e) \cap \Delta]^v_b \neq \emptyset \text{ (when } k < l); \\ [R(e)]^D_v \neq \emptyset \text{ (when } k = l \neq L) \end{matrix}\right\}\right]^v_S\right) \tag{6}$$
$$(\Delta \subseteq E).$$

Operator 4 ($\psi_{R(E)-\{R(e)\}}(R(e))$, analytic form). Given $e = \langle u, v, l\rangle \in E$ $(1 < l < L)$ and the collection of ρ-path edge-sets $R(E)$. Give the operator $\psi_{R(E)-\{R(e)\}}(R(e))$ as the "*analytic form*": $\psi_{R(E)-\{R(e)\}}(R(e))$ modifies $R(e)$ into an edge set $ES1$, s.t. $ES1 \subseteq R(e)$, $\mathfrak{Y}(ES1) = true$ and $\mathfrak{Y}(ES2) = false$ $(ES2 \subseteq R(e))$ for each possible $ES2 \supset ES1$, where

---

[9] To specify the context when necessary—i.e., a given property in G and its peer in another graph say G'—we use the indicators (*something* of G) and (*something* of G') respectively.

[10] For brevity, if more than one "(of G)" are intended for a bundle of attributes in the same context (e.g., G), we just use one outermost "(of G)". Let's take the term $\left(\chi^D_{R(E)}(ESS1) \text{ of } G'\right)$ as an example. The $ESS1, R(E)$ within the term each actually means $(ESS1 \text{ of } G'), (R(E') \text{ of } G')$, since they are directly enveloped in the context indicator "(of E')". Otherwise, if we mean $(ESS1 \text{ of } G), (R(E) \text{ of } G)$, the term should be written as $\left(\chi^D_{(R(E) \text{ of } G)}(ESS1 \text{ of } G) \text{ of } G'\right)$.

$$\mathfrak{Y}(\Delta)=_{\text{def}}(\Delta = \left[\left\{ e' = \langle a,b,k\rangle \in \Delta \;\middle|\; \begin{array}{c} \boldsymbol{B}\neq\emptyset, \text{ where} \\ \boldsymbol{A}=\chi^{b}_{R(E)}(\{\langle x,y,i\rangle\in E | e'\in[R(\langle x,y,i\rangle)\cap\lambda(b)]^{b}_{y}\}\cup\{e'\}) \\ \text{and } \boldsymbol{B}=\chi^{u}_{R(E)}(\{\langle c,d,j\rangle\in\boldsymbol{A}|\{e,e'\}\subseteq[R(\langle c,d,j\rangle)\cap\boldsymbol{A}]^{b}_{d}\}) \end{array} \right\}\right]^{D}_{v}) \quad (7)$$
$$(\Delta\subseteq E).$$

The following theorems provide guarantee that, the two different types of operator definitions coincide. In other words, the intended algorithmic operations underpinning the proposed ZH algorithm are all accurately and uniquely determined.

Theorem 2 (Equivalence & uniqueness, Operator 3). *The "analytic form" and the "procedural form" define the same operator. In other words, there exists only one unique edge set that fulfills the "analytic form" (or the "procedural form") of Operator 3.*

Proof. Let $\Delta=[\Delta]^{v}_{S}\subseteq ES$ be an edge set that fulfills its "analytic form", and let $\Delta'=[\Delta']^{v}_{S}\subseteq ES$ be the result of its "procedural form".

(1) If $\Delta\subset\Delta'$. According to the computation of procedural form, for some $\langle a,b,k\rangle\in(\Delta'-\Delta)\subseteq ES$, we must have $\begin{cases}[R(e)\cap\Delta]^{v}_{b}\neq\emptyset \text{ (when } k<l)\\ [R(e)]^{D}_{v}\neq\emptyset \text{ (when } k=l\neq L)\end{cases}$. This contradicts with the definition of Δ.

(2) If $\Delta'\subset\Delta$. Any $\langle a,b,k\rangle\in(\Delta-\Delta')\subseteq ES$, will be deleted according to the computation of procedural form, since $\begin{cases}[R(e)\cap\Delta]^{v}_{b}=\emptyset \text{ (when } k<l)\\ [R(e)]^{D}_{v}=\emptyset \text{ (when } k=l\neq L)\end{cases}$.

(3) Any other case, $\Delta\subset\Delta\cup\Delta'$. This will violate the definition of analytic form. □

Theorem 3 (Equivalence & uniqueness, Operator 4). *The "analytic form" and the "procedural form" define the same operator. In other words, there exists only one unique edge set that fulfills the "analytic form" (or the "procedural form") of Operator 4.*

Proof. Let $\Delta=[\Delta]^{D}_{v}\subseteq R(e)$ be an edge set that fulfills its "analytic form", and let $\Delta'=[\Delta']^{D}_{v}\subseteq R(e)$ be the result of its "procedural form".

(1) If $\Delta\subset\Delta'$. According to the computation of procedural form, for some $\langle a,b,k\rangle\in(\Delta'-\Delta)\subseteq R(e)$, we must have $\boldsymbol{A}=\chi^{b}_{R(E)}(\{\langle x,y,i\rangle\in E|e'\in[R(\langle x,y,i\rangle)\cap\lambda(b)]^{b}_{y}\}\cup\{e'\})\neq\emptyset$ and $\boldsymbol{B}=\chi^{u}_{R(E)}(\{\langle c,d,j\rangle\in\boldsymbol{A}|\{e,e'\}\subseteq[R(\langle c,d,j\rangle)\cap\boldsymbol{A}]^{b}_{d}\})\neq\emptyset$. This contradicts with the definition of Δ.

(2) If $\Delta'\subset\Delta$. Any $\langle a,b,k\rangle\in(\Delta-\Delta')\subseteq R(e)$, will be deleted according to the computation of procedural form, since $\boldsymbol{A}=\chi^{b}_{R(E)}(\{\langle x,y,i\rangle\in E|e'\in[R(\langle x,y,i\rangle)\cap\lambda(b)]^{b}_{y}\}\cup\{e'\})=\emptyset$ and $\boldsymbol{B}=\chi^{u}_{R(E)}(\{\langle c,d,j\rangle\in\boldsymbol{A}|\{e,e'\}\subseteq[R(\langle c,d,j\rangle)\cap\boldsymbol{A}]^{b}_{d}\})=\emptyset$.

(3) Any other case, $\Delta\subset\Delta\cup\Delta'$. This will violate the definition of analytic form. □

8 Proof of Theorem 4

Theorem 4. *The cost of the ZH algorithm can be* $O(|E|^{9})$.

Proof. Each size of $\lambda(v)$ and $R(e)$ is no more than $|E|$. Moreover, $|R(E)|$ is no more than $|E|$.

The cost for computing $\chi^{v}_{R(E)}(ES)$ can be $O(|E|^3)$ and the cost for computing $\psi_{R(E)-\{R(e)\}}(R(e))$ can be $O(|E|^6)$, hence the cost of step 2 of the ZH algorithm can be $O(|E|^7)$.

Step 2 is the most expensive statement in the ZH algorithm. Each iteration of step 2 will prune at least one edge in $R(\langle u,v,l\rangle)$, and the number of edges each in $R(\langle u,v,l\rangle)$ and $\lambda(v)$ is no more than $|E|$. The number of $R(e)$ is $|R(E)|$. So, the cost of step 2 and step 3 can be $|E| * |R(E)| * O(|E|^7)$.

Overall, the cost of the ZH algorithm can be $O(|E|^9)$, which is a polynomial function of $|E|$.□

9 Proof of Theorem 5

Theorem 5. *If G contains a σ-path, then the compact kernel of G is not empty.*

Proof. Let $P = v_0 - v_1 - v_2 - \cdots - v_L$ be a σ-path in G, where $v_0 = S$ and $v_L = D$. By Definition 2, $[P]^{v_h}_{v_0} \subseteq \lambda(v_h)$ $(1 \le h \le L)$, and for $\langle v_{l-1}, v_l, l\rangle \in P$ $(1 \le l \le L)$ we have $\langle v_{l-1}, v_l, l\rangle \in \lambda(v_l) \cap \lambda(v_{l+1}) \cap \cdots \cap \lambda(D)$. Thus, after the execution of step 1 of the ZH algorithm, we have $[P]^{v_L}_{v_l} \subseteq R(\langle v_{l-1}, v_l, l\rangle)$ $(1 \le l \le L)$. After step 2, we still have $[P]^{v_L}_{v_l} \subseteq R(\langle v_{l-1}, v_l, l\rangle)$ $(1 \le l \le L)$. Step 3 can not prune any path in $R(\langle v_{l-1}, v_l, l\rangle)$. This will ensure that $P \subseteq \chi^{D}_{R(E)}(\lambda(D))$. Hence, $\chi^{D}_{R(E)}(\lambda(D)) \neq \emptyset$. □

10 Proof of Lemma 1

Lemma 1. *Let $G =< V, E, S, D, L, \lambda >$ be the input to the PA and there is no multi-in-degree vertex at stage 1,2,…, $L-1$ in G (see Figure 5). After applying the PA on G, if $[ES1_{sub}]^{D}_{S}[L\colon L] = ES2$, then $(\forall \langle w, D, L\rangle \in [ES1_{sub}]^{D}_{S})(\exists\ \sigma - \text{path}\ S - \cdots - w - D \subseteq ES1_{sub})$.*

Proof. $[ES1_{sub}]^{D}_{S}[L\colon L] = ES2 \neq \emptyset$ implies $ES1 = \chi^{D}_{R(E)}(ES1) \neq \emptyset$. Then, for $\langle a_{L-1}, D, L\rangle \in ES2$, there exists $\langle a_{L-2}, a_{L-1}, L-1\rangle \in ES1$ such that $\langle a_{L-1}, D, L\rangle \in R(\langle a_{L-2}, a_{L-1}, L-1\rangle) \cap \chi^{D}_{R(E)}(ES1)$ by the definition of Operator 3. According to the computation of $\psi_{R(E)-\{R(\langle a_{L-2}, a_{L-1}, L-1\rangle)\}}(R(\langle a_{L-2}, a_{L-1}, L-1\rangle))$, when deciding "$\langle a_{L-1}, D, L\rangle \in R(\langle a_{L-2}, a_{L-1}, L-1\rangle)$", we have

$$\boldsymbol{A} = \chi^{D}_{R(E)}\left(\left\{\langle x,y,i\rangle \in E \middle| \begin{array}{c}\langle a_{L-1}, D, L\rangle \in \\ [R(\langle x,y,i\rangle) \cap \lambda(D)]^{D}_{y}\end{array}\right\} \cup \{\langle a_{L-1}, D, L\rangle\}\right) \neq \emptyset. \tag{8}$$

By the definition of Operator 3 and by the fact that no multi-in-degree vertex can be found in G from stage 1 to stage $L-1$, only one single preceding edge of stage $l-1$ can be found for each edge of stage l $(2 \le l \le L)$ in $\boldsymbol{A}$. Thus, the set $\boldsymbol{A}$ is uniquely determined as the path $S - \cdots - a_{L-2} - a_{L-1} - D$, which must be a σ-path and $S - \cdots - a_{L-2} = [ES1]^{a_{L-2}}_{S}$.

The above discussion based on Operator 3 and on the graph structure also implies that, (i) each path $S - \cdots - D \subseteq ES1$ must be a σ-path and (ii) each $S - \cdots - D \subseteq [ES1_{sub}]^{D}_{S} \subseteq ES1$ must also be a σ-path. □

11 Proof of Lemma 2

Lemma 2. *Given the mathematical induction hypothesis H1 that the PA can make a correct decision for any G' that $f(G') < m$ $(m > 0)$. Let $G =< V, E, S, D, L, \lambda >$ be the input to the PA, $f(G) = m$, the vertex v of stage l $(1 < l < L)$ be a multi-in-degree vertex, and there exists no multi-in-degree vertex (except D) above stage l (see Figure 6(a)). After applying the PA on G, if $[ES1_{sub}]_S^D[L\colon L] = ES2$, then $(\forall \langle w, D, L\rangle \in [ES1_{sub}]_S^D)(\exists\ \sigma - \text{path}\ S - \cdots - w - D \subseteq ES1_{sub})$.*

11.1 Sketch of the proof

To prove Lemma 2, we need to construct a graph $G' =< V', E', S, D, L, \lambda' >$, such that: (1) $f(G') < f(G)$, and G' satisfies Definition 3(b); (2) if $([ES1_{sub}]_S^D[L\colon L]\ \text{of}\ G)$[11] $= (ES2\ \text{of}\ G)$ for G, then $([ES1_{sub}]_S^D[L\colon L]\ \text{of}\ G') = (ES2\ \text{of}\ G')$ for G'; (3) if $(SP\ \text{of}\ G') \subseteq (ES1_{sub}\ \text{of}\ G')$ is a solution demanded by the PA for G', then some solution $(SP\ \text{of}\ G) \subseteq (ES1_{sub}\ \text{of}\ G)$ demanded by the PA must can be found for G.

Taking such a properly constructed G', Lemma 2 can get proved, by getting divided into the following Claim 1,2,3,4,5. Implicitly, these claims share the same context of Lemma 2.

Claim 1. *$f(G') < f(G)$.*

Claim 2 (Informal). *If $(ES1\ of\ G) \neq \emptyset$, we have $(ES1\ of\ G') \neq \emptyset$; $(ES1\ of\ G')$ contains essentially no less edges than $(ES1\ of\ G)$.* (the statement is slightly adjusted here, which will be more accurately restated later)

Claim 3. *G' satisfies Definition 3(b).*

Claim 4 (Picking out σ-paths). *If $\langle a, b, h\rangle \in (\lambda_{sub}(y)\ of\ G)$ $(\langle a, b, h\rangle \in E$, $\langle y, D, L\rangle \in E$, $1 < h \leq l)$, there exists a σ-path that traverses $\langle a, b, h\rangle$ and $\langle y, D, L\rangle$ in G.*

Claim 5. *If $([ES1_{sub}]_S^D[L\colon L]\ of\ G) = (ES2\ of\ G)$, then $(\forall \langle w, D, L\rangle \in ([ES1_{sub}]_S^D\ of\ G))\big)\ (\exists\ \sigma - \text{path}\ S - \cdots - w - D \subseteq (ES1_{sub}\ of\ G))$.*

11.2 The construction of a less equivalent G_1 and the proof of Claim 1,2,3,4

For the multi-in-degree vertex $v \in V_l$ $(1 < l < L)$ specified by Lemma 2, we have $d^+(v) > 0$ (see Definition 3(b) (item 1)) and $d^-(t) = \cdots = d^-(w) = 1$ for each path from v to D like $v - t - \cdots - w - D$[12]. Obviously $d^-(v) = 2$ (see Definition 3(b)), and let's assume $\langle u_1, v, l\rangle$ and $\langle u_2, v, l\rangle$ are just the two edges ending at v, as shown in Figure 6(a).

By Definition 3(b) (item 2), we have $2 \leq l \leq L - 2$ (recall that $L \geq 5$ by Definition 1).

Based on G, we can construct a new graph $G_1 =< \widehat{V_1}$[13]$, E_1, S, D, L, \lambda_1 >$ as follows, by

[11] As introduced in Section 6, the notation specifies the contextual graph for a given property.

[12] The path is allowed to be shorter than 3 edges. The introduction of the additional vertex "t" just helps the illustration but is not a must. The same is with the below.

[13] When defining a graph G_n $(n \in \mathbb{N})$, we use $\widehat{V_n}$ to represent the set of all vertices in G_n, instead of V_n. In this way, $\widehat{V_n}$ can be distinguished from the "V_n" in Definition 1 (recall that V_n represents the set of all vertices at a specified stage $n \in \{0, ..., L\}$).

"splitting" the multi-in-degree vertex v.

To define $\widehat{V_1}$ and E_1, we delete all paths $u_i - v - \cdots\cdots - D$ $(i = 1,2)$ in G and add two new paths $u_i - v_i - t_i - \cdots - r_i - w_i - D$ $(i = 1,2)$. Keep all the rest vertices and edges unchanged. We then get the structure of an L −stage graph G_1, as shown in Figure 6(b).

To define λ_1 (i.e., $(\lambda(x)$ of $G_1)$ for $x \in \widehat{V_1}$):

(i) For $x = D$, let $(\lambda(D)$ of $G_1) = E_1$.

(ii) For $x \in V - \{v_1, v_2, t_1, t_2, ..., w_1, w_2, D\}$, let $(\lambda(x)$ of $G_1) = (\lambda(x)$ of $G)$.

(iii) For $x \in \{v_1, v_2, t_1, t_2, ..., w_1, w_2\}$, the following **radical expansion** of labels is done:

- For $x \in \{v_1, v_2\}$, set $(\lambda(v_i)$ of $G_1) = \{\langle u_i, v_i, l\rangle\} \cup \begin{pmatrix} (\lambda(v) \text{ of } G)[1\colon l-1] \\ \cap\, (\lambda(u_i) \text{ of } G)[1\colon l-1] \end{pmatrix}$ $(i = 1,2)$.
- For x on $t_1 - \cdots - w_1$ and $t_2 - \cdots - w_2$, set $(\lambda(t_i)$ of $G_1) = (\lambda(v_i)$ of $G_1) \cup \{\langle v_i, t_i, l+1\rangle\}$, ..., $(\lambda(w_i)$ of $G_1) = (\lambda(v_i)$ of $G_1) \cup (v_i - t_i - \cdots - r_i - w_i)$ $(i = 1,2)$.

The above "split" of v won't damage the existence of the original σ-paths. Indeed, the radical expansion makes it easy to confirm the computation and the result of $R(e)$ after applying $\psi_{R(E)-\{R(e)\}}(R(e))$, when G_1 is the input to the PA.

However, since $(\lambda(x)$ of $G_1)$ $(x \in \widehat{V_1})$ seems to contain more edges in essence than its peer in G, if P is a σ-path in G_1, maybe no σ-path corresponding to P exists in G. Nevertheless, in the current situation of Claim 1,2,3,4, this won't cause troubles. Moreover, in Claim 5, it will be proved that, if there is a σ-path as claimed by step 3 of the PA for the "smaller" graph, then there must exist a σ-path as claimed by step 3 of the PA for G. Hence, the constraints posed by $(ES1_{sub}$ of $G_1)$ (if non-empty) manage to hold back the undesired solutions introduced by the radical expansion (and the expansion now actually becomes **conservative**).

Subsequently, the construction of G_1 is completed.

Claim 1 (for G_1). *$f(G_1) < f(G)$.*

Proof. Since v is the multi-in-degree vertex that appears at stage l $(1 < l < L)$ and no multi-in-degree vertex (except D) can be found above stage l, we have

$$\begin{aligned} &\sum\nolimits_{x\in(V_l \text{ of } G_1)} (d^-(x) - 1) \\ =\ &\sum\nolimits_{x\in(V_l \text{ of } G_1)-\{v_1,v_2\}} (d^-(x) - 1) + (d^-(v_1) - 1) + (d^-(v_2) - 1) \\ \le\ &\sum\nolimits_{x\in(V_l \text{ of } G)-\{v\}} (d^-(x) - 1) + (d^-(v) - 1) - 1 \\ =\ &\sum\nolimits_{x\in(V_l \text{ of } G)} (d^-(x) - 1) - 1\,. \end{aligned} \tag{9}$$

Therefore, $f(G_1) < f(G)$. □

Claim 2 (for G_1). *If $\left(\chi^D_{R(E)}(ES1)\ of\ G\right) \neq \emptyset$, we have*

$$(ES1\ of\ G_1) \supseteq \tag{10}$$

$$\left(\begin{array}{c}\Big((ES1\ of\ G)-\{e|e\in u_i-v-\cdots\cdots-D\subseteq E,i\in\{1,2\}\}\Big)\\ \cup\left\{e\middle|\begin{array}{l}e\in u_i-v_i-\cdots-w_i-D\subseteq E_1,\\(R(\langle u_i,v,l\rangle)\ of\ G)\neq\emptyset,i\in\{1,2\}\end{array}\right\}\end{array}\right)\neq\emptyset.$$

Proof. For every $\langle r,s,k\rangle$ and $\langle o,p,h\rangle$ $(1\leq k<h\leq L)$ in G: (i) if the initial ρ-path edge-set $(R_0(\langle r,s,k\rangle)$ of $G)$ contains $\langle o,p,h\rangle$, there must exist some e_1 and e_2 in G_1, such that $e_2\in(R_0(e_1)$ of $G_1)$; (ii) if the $(R(\langle r,s,k\rangle)$ of $G)$[14] $\in(R(E)$ of $G)$ computed by the ZH algorithm contains $\langle o,p,h\rangle$, according to the radical expansion of G_1, there must exist e_1 and e_2 in G_1, such that the $(R(e_1)$ of $G_1)\in(R(E)$ of $G_1)$ computed by the ZH algorithm contains e_2.

If the above $\langle r,s,k\rangle$ and $\langle o,p,h\rangle$ are associated with the vertices involved in $[E]_v^D[l+1\colon L]$, then e_1 and e_2 are associated with the vertices (of the corresponding stages) involved in $[E_1]_{v_1}^D[l+1\colon L]\cup[E_1]_{v_2}^D[l+1\colon L]$, otherwise $e_1=\langle r,s,k\rangle$ and $e_2=\langle o,p,h\rangle$. For instance, when $\langle o,p,h\rangle\in v-\cdots\cdots-D\subseteq E$ and $k<l$, $\langle o,p,h\rangle\in(R(\langle r,s,k\rangle)$ of $G)$ implies $\langle u_i,v,l\rangle\in(R(\langle r,s,k\rangle)$ of $G)$ (for some $i\in\{1,2\}$); then by the radical expansion, we can have $\langle u_i,v_i,l\rangle\in(R(\langle r,s,k\rangle)$ of $G_1)$ and further we can deduce that there exists an edge $e_2\in v_i-\cdots-w_i-D\subseteq E_1$ of stage h in G_1 such that $e_2\in(R(\langle r,s,k\rangle)$ of $G_1)$.

More discussions on $(R(E)$ of $G_1)$ and the detailed renaming rules for the above e_1 and e_2, if needed, are provided in Section 13.1.

With the above clarification of $(R(E)$ of $G_1)$, then by the definition of Operator 3, we can hence obtain

$$\left(\chi_{R(E)}^D(\lambda(D))\text{ of }G_1\right)\supseteq\left(\begin{array}{c}\left(\chi_{R(E)}^D(\lambda(D))\text{ of }G\right)-\\ \{e|e\in u_i-v-\cdots\cdots-D\subseteq E,i\in\{1,2\}\}\end{array}\right).\tag{11}$$

Note that, we have $\langle u_i,v,l\rangle\in\left(\chi_{R(E)}^D(\lambda(D))\text{ of }G\right)$ when $(R(\langle u_i,v,l\rangle)$ of $G)\neq\emptyset$ $(i\in\{1,2\})$, because $\emptyset\neq\left(\chi_{R(E)}^D\left(\begin{array}{c}\boldsymbol{A}[l+1\colon L]\cup\\ \{\langle u_i,v,l\rangle\}\cup\boldsymbol{B}\end{array}\right)\text{ of }G\right)\subseteq\left(\chi_{R(E)}^D(\lambda(D))\text{ of }G\right)$ (where $(\boldsymbol{A}$ of $G),(\boldsymbol{B}$ of $G)$ are the sets computed when deciding to preserve some $\langle w,D,L\rangle\in E$ in $(R(\langle u_i,v,l\rangle)$ of $G)$ by Operator 4, see step 2 of $ZH\setminus step4$) by bottom-up checking the edges in the definition of Operator 3 and by leveraging the fact that $(\boldsymbol{A}$ of $G)\subseteq(\lambda(D)$ of $G)$.

Further note that, $(R(\langle u_i,v,l\rangle)$ of $G)\neq\emptyset$ $(i\in\{1,2\})$ implies $\langle w,D,L\rangle\in(R(\langle u_i,v,l\rangle)$ of $G)$. That further implies $\langle w_i,D,L\rangle\in(R(\langle u_i,v_i,l\rangle)$ of $G_1)$, and hence $v_i-t_i-\cdots-r_i-w_i-D\subseteq(R(\langle u_i,v_i,l\rangle)$ of $G_1)$ by the radical expansion.

Subsequently, by the definition of Operator 3, we can have

$$\left(\chi_{R(E)}^D(\lambda(D))\text{ of }G_1\right)\supseteq\left\{e\middle|\begin{array}{l}e\in u_i-v_i-\cdots-w_i-D\subseteq E_1,\\(R(\langle u_i,v,l\rangle)\text{ of }G)\neq\emptyset,i\in\{1,2\}\end{array}\right\}.\tag{12}$$

Summarizing all above discussions, we now obtain

$$\left(\chi_{R(E)}^D(\lambda(D))\text{ of }G_1\right)\supseteq\tag{13}$$

[14] Unless otherwise specified, $R(e)$ refers to the stable one after step 2 of $ZH\setminus step4$, since it is the minimum.

$$\left(\begin{array}{c}\left(\left(\chi^{D}_{R(E)}(\lambda(D))\text{ of }G\right)-\left\{e\middle|e\in u_i-v-\cdots\cdots-D\subseteq E,i\in\{1,2\}\right\}\right)\\ \cup\left\{e\middle|\begin{array}{l}e\in u_i-v_i-\cdots-w_i-D\subseteq E_1,\\(R(\langle u_i,v,l\rangle)\text{ of }G)\neq\emptyset,i\in\{1,2\}\end{array}\right\}\end{array}\right).$$

As a result, $\left(\chi^{D}_{R(E)}(\lambda(D))\text{ of }G\right)\neq\emptyset$ would imply $\left(\chi^{D}_{R(E)}(ES1)\text{ of }G_1\right)=(ES1\text{ of }G_1)=\left(\chi^{D}_{R(E)}(\lambda(D))\text{ of }G_1\right)\neq\emptyset$. □

Claim 3 (for G_1). *G_1 satisfies Definition 3(b).*

Proof. Recall that G fulfills Definition 3(b).

It is easy to check item-by-item that, the "split" of v won't violate Definition 3(b) for G_1, since no multi-in-degree vertex except D can be found above stage l in both G and G_1. □

Claim 4 (Picking out σ-paths). *If $\langle a,b,h\rangle\in(\lambda_{sub}(y)\ of\ G)$ ($\langle a,b,h\rangle\in E$, $\langle y,D,L\rangle\in E$, $1<h\leq l$), there exists a σ-path that traverses $\langle a,b,h\rangle$ and $\langle y,D,L\rangle$ in G.*

Claim 4 serves as a key tool to help us "split" the system invariant $ES1_{sub}$ for our constructed "smaller" graph. Claim 4 can be broken down to the following sub-claims (Claim 4a,4b,4c), with regard to the varied locations of the edge $\langle a,b,h\rangle$. The key to its proof is that, the computed ($\boldsymbol{A}$ of G) by Operator 4 for $\langle w,D,L\rangle\in(R(e)\text{ of }G)$ fixes both e and $\langle y,D,L\rangle$, and hence can be utilized for building ($ES1_{sub}$ of G_1) and infer the desired σ-path by our mathematical induction on $f(G)$.

Claim 4a. *Given $\langle a,b,h\rangle\in E$ ($1<h<l-1$), $\langle w,D,L\rangle\in v-\cdots-D\subseteq E$. If $\langle w,D,L\rangle\in(R(\langle a,b,h\rangle)\ of\ G)$, there exists a σ-path that traverses $\langle a,b,h\rangle$ and $\langle w,D,L\rangle$ in G.*

Proof. Let $e=\langle a,b,h\rangle$.

The path from v to w in G is uniquely determined, by the structure of G. Let it be denoted by $v-t-\cdots-r-w$.[15]

First note the following facts:

- By the sets ($\boldsymbol{A}$ of G) $\neq\emptyset$, ($\boldsymbol{B}$ of G) $\neq\emptyset$ computed for deciding "$\langle w,D,L\rangle\in(R(e)\text{ of }G)$" by Operator 4 in step 2 of $ZH\setminus step4$, it can be inferred that $v-t-\cdots-r-w-D\subseteq\boldsymbol{A}$ is a ω-path.
- It also can be observed that, there exists a non-empty set $J\subseteq\{1,2\}$, such that for each $j\in J$: (i) $\langle u_j,v,l\rangle\in\left((\boldsymbol{A}\cap R(e))\text{ of }G\right)$, because ($R(e)$ of G) must contain a reachable path which traverses $\langle w,D,L\rangle$ via the vertex v when applying Operator 3 for computing ($\boldsymbol{A}$ of G); (ii) $\langle w,D,L\rangle\in(R(\langle u_j,v,l\rangle)\text{ of }G)$, because $\langle u_j,v,l\rangle\in(\boldsymbol{A}\text{ of }G)$ and $\langle w,D,L\rangle$ is the unique edge of stage L in ($\boldsymbol{A}$ of G).
- The radical expansion forces each label on $v_j-t_j-\cdots-r_j-w_j-D$ ($j\in J$) in G_1 to contain $(\lambda(v_j)[1{:}\,l-1]\text{ of }G_1)=\left(\left(\lambda(u_j)\cap\lambda(v)\right)[1{:}\,l-1]\text{ of }G\right)=\left((\lambda(u_j)\cap\lambda(v_j)\right)[1{:}\,l-1]\text{ of }G_1\Big)$ as a subset. Subsequently, "$\langle u_j,v,l\rangle\in(R(e)\text{ of }G)$" implies

[15] The definition of paths $u_i-v_i-t_i-\cdots-r_i-w_i-D\subseteq E_1$ ($i=1,2$) might introduce a mild abuse of notation about the "t_i,...,r_i,w_i" on the paths and those "t,...,r,w" on an arbitrarily designated $v-t-\cdots-r-w\subseteq E$. There is actually no direct correspondence between them, although the same letters "t,...,r,w" are shared.

"$\langle u_j, v_j, l\rangle \in (R(e) \text{ of } G_1)$" (by the radical expansion of $(\lambda(v_j) \text{ of } G_1)$) and hence "$v_j - t_j - \cdots - r_j - w_j - D \subseteq (R(e) \text{ of } G_1)$" (by the radical expansion of $(\lambda(t_j) \text{ of } G_1), \ldots, (\lambda(w_j) \text{ of } G_1)$); "$\langle w, D, L\rangle \in (R(\langle u_j, v, l\rangle) \text{ of } G)$" implies "$\langle w_j, D, L\rangle \in (R(\langle u_j, v_j, l\rangle) \text{ of } G_1)$" and hence "$v_j - t_j - \cdots - r_j - w_j - D \subseteq (R(\langle u_j, v_j, l\rangle) \text{ of } G_1)$". For detailed argument, if needed, see the renaming rules and the "transit" technique discussed in Section 13.1 for $(R(E) \text{ of } G_1)$.

Back to the computation performed on G which decides "$\langle w, D, L\rangle \in (R(e) \text{ of } G)$". Guided by the corresponding edge set $(\boldsymbol{A} \text{ of } G), (\boldsymbol{B} \text{ of } G)$ that is involved in Operator 4, some appropriate $ES1_{sub}$ for the "smaller" graph G_1 can be constructed as follows.

If we choose in a "single-plank bridge" way that

$$(ES\_temp \text{ of } G_1) =_{def} \left(\left((\boldsymbol{A}[1\!:\!l-1] \text{ of } G) \cup \left\{ e' \middle| \begin{matrix} e' \in u_i - v_i - \cdots - w_i - D \subseteq E_1, \\ \langle u_i, v, l\rangle \in (\boldsymbol{A} \text{ of } G), i \in \{1,2\} \end{matrix} \right\} \right) - \{e' \in E | e' \neq e \text{ is an edge of stage } h\} \right), \tag{14}$$

then we can still have $\left(\chi^D_{R(E)}(ES\_temp) \text{ of } G_1\right) \neq \emptyset$, despite our removal of edges of stage h. That is because the computation of $\left(\chi^D_{R(E)}(ES\_temp) \text{ of } G_1\right)$ is essentially the same as the computation of $(\psi_{R(E)-\{R(e)\}}(R(e)) \text{ of } G)$ when deciding "$\langle w, D, L\rangle \in (R(e) \text{ of } G)$". This is straightforward by the radical expansion—according to the above discussion, for each $\varepsilon \in \left(\left(\boldsymbol{A} - \left\{ e' \in E \middle| \begin{matrix} e' \neq e \text{ is an} \\ \text{edge of stage } h \end{matrix} \right\} \right) \text{ of } G \right)$ ($\varepsilon \neq \langle w, D, L\rangle$), there exists $\hat{\varepsilon} \in E_1$ such that $\langle w_i, D, L\rangle \in (R(\hat{\varepsilon}) \text{ of } G_1)$ (where $i \in \{1,2\}$ such that $\langle u_i, v, l\rangle \in (\boldsymbol{A} \text{ of } G)$). As shown in Figure 8. Refer to Section 13.2 for more detailed discussion, if needed.

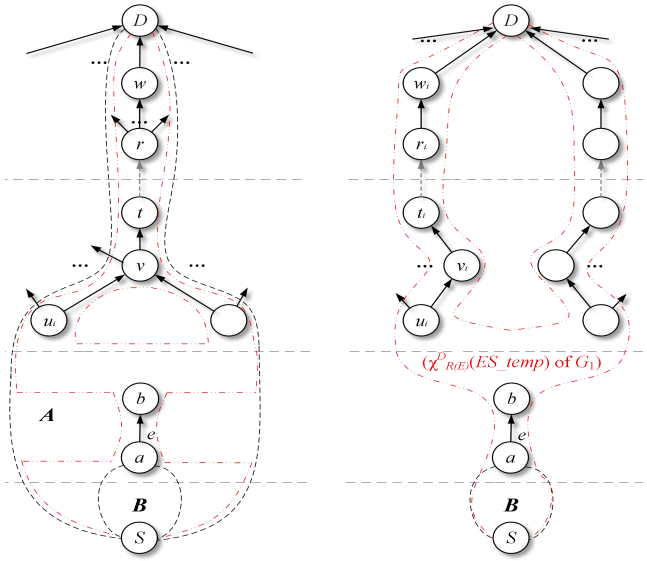


Figure 8: Illustration of Claim 4a

Since $\left(\chi^D_{R(E)}(ES\_temp) \text{ of } G_1\right) \neq \emptyset$, there must exist some $\langle \alpha, \beta, 2\rangle \in \left(\chi^D_{R(E)}(ES\_temp) \text{ of } G_1\right)$ such that $H = \beta - \cdots - D \subseteq ([R(\langle \alpha, \beta, 2\rangle) \cap$

$\chi^{D}_{R(E)}(ES\_temp)\big]^{D}_{\beta}$ of $G_1\big) \neq \emptyset$, by the definition of Operator 3. Assume $\langle u_1, v_1, l\rangle \in H$. Besides, if $|(\chi^{D}_{R(E)}(ES\_temp)[L\colon L]$ of $G_1)| = 2$, there must exist a path $H' = x - \cdots - D \subseteq (\chi^{D}_{R(E)}(ES\_temp)$ of $G_1)$ such that x is the unique non-sink vertex on both H, H'; otherwise, assume $H' = \emptyset$.

Then, we can choose

$$(ES2 \text{ of } G_1) =_{def} (\chi^{D}_{R(E)}(ES\_temp)[L\colon L] \text{ of } G_1), \tag{15}$$

$$(ES1_{sub} \text{ of } G_1) =_{def} (S - \alpha - \beta) \cup H \cup H'. \tag{16}$$

If define $(\lambda_{sub}(w_i)$ of $G_1) =_{def} (ES1_{sub}$ of $G_1) \cap (\lambda(w_i)[1\colon 1] \cup \{e \in \lambda(w_i)[2\colon L] | \langle w_i, D, L\rangle \in R(e)\})$ $(i = 1{,}2)$, $(ES1_{sub}$ of $G_1)$ obeys criteria (i),(ii),(iv). Criterion (iii) is apparently obeyed, since we can assume that no σ-path contained in $(ES1_{sub}$ of $G_1)$ exists, otherwise Claim 4a is proved. Criterion (v) is also obeyed, since $\langle w_1, D, L\rangle \in (R(e)$ of $G_1)$ for some $e \in [(S - \alpha - \beta) \cup H]^{x}_{S}$ while $\langle w_2, D, L\rangle \in (R(e'$ of $G_1)$ for some $e' \in [(S - \alpha - \beta) \cup H]^{x}_{S} - \{e\}$; otherwise, just alliteratively define $(ES2$ of $G_1) =_{def} \{\langle w_1, D, L\rangle\}$ and $(ES1_{sub}$ of $G_1) =_{def} (S - \alpha - \beta) \cup H$. Meanwhile, we can assume that no σ-path contained in $(ES1_{sub}$ of $G_1)$ exists, and hence $(\lambda_{sub}(w_i)$ of $G_1)$ $(i = 1{,}2)$ has no obligation to be set to include the whole $[(S - \alpha - \beta) \cup H]^{x}_{S}$.

It can be further straightforwardly observed that $([ES1_{sub}]^{D}_{S}[L\colon L]$ of $G_1) = (ES2$ of $G_1) \neq \emptyset$, by the definition of $(ES\_temp$ of $G_1)$ and by the fact that $\left[\begin{pmatrix}(\chi^{D}_{R(E)}(ES\_temp)[3\colon L] \text{ of } G_1)\\ \cup (S - \alpha - \beta)\end{pmatrix}\right]^{D}_{S}[L\colon L] = \left[(\chi^{D}_{R(E)}(ES\_temp) \text{ of } G_1)\right]^{D}_{S}[L\colon L] \subseteq \left\{\begin{matrix}\langle w_1, D, L\rangle,\\ \langle w_2, D, L\rangle\end{matrix}\right\}$.

Then, by our mathematical induction hypothesis (H1), we can infer that there exists some σ-path $SP = S - \cdots - a - b - \cdots - u_i - v_i - \cdots - w_i - D \subseteq (ES1_{sub}$ of $G_1)$ $(i \in \{1{,}2\})$ in G_1 by the PA. Note that $SP[1\colon l-1] \cup (u_i - v - \cdots - w - D) \subseteq (\boldsymbol{A}$ of $G)$, and hence straightforwardly $SP[1\colon l-1] \cup (u_i - v - \cdots - w - D)$ is a σ-path in G by Operator 3. □

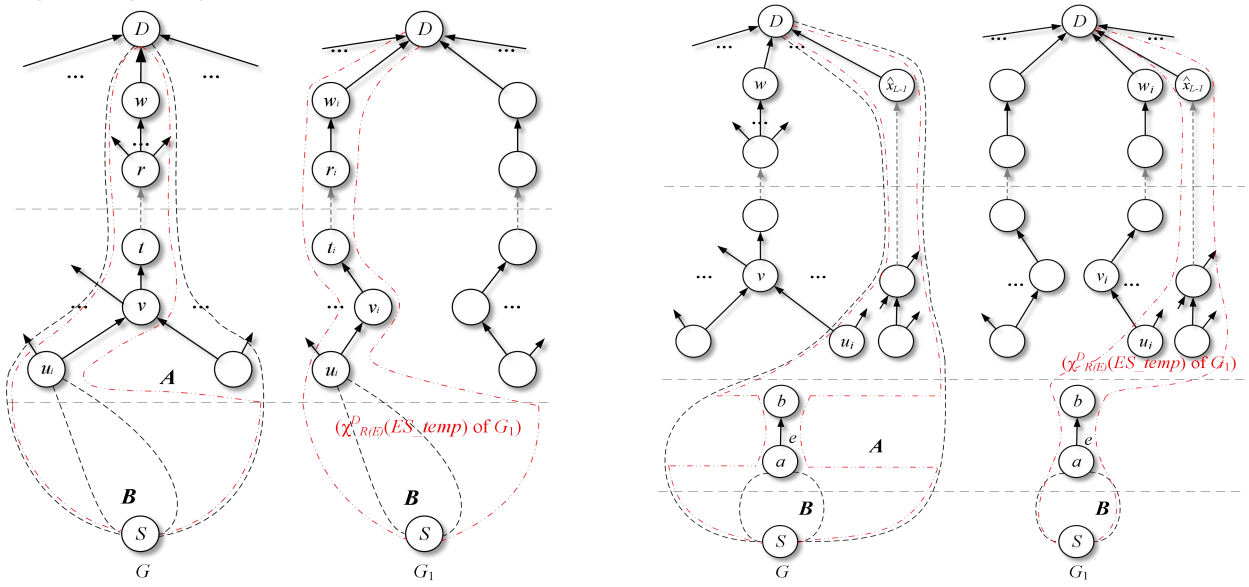


Figure 9: Illustration of Claim 4b, Claim 4c

Claim 4b. *Given* $\langle a,b,h\rangle \in E$ *(*$h>1$*,* $l-1\le h\le l$*),* $\langle w,D,L\rangle \in v-\cdots-D\subseteq E$*. If* $\langle w,D,L\rangle \in (R(\langle a,b,h\rangle)$ *of* G*), there exists a* σ*-path that traverses* $\langle a,b,h\rangle$ *and* $\langle w,D,L\rangle$ *in* G*.*

Claim 4c. *Given* $\langle a,b,h\rangle \in E$ *(*$1<h\le l$*),* $\langle \hat{x}_{L-1},D,L\rangle \notin v-\cdots-D\subseteq E$*. If* $\langle \hat{x}_{L-1},D,L\rangle \in (R(\langle a,b,h\rangle)$ *of* G*), there exists a* σ*-path that traverses* $\langle a,b,h\rangle$ *and* $\langle \hat{x}_{L-1},D,L\rangle$ *in* G*.*

Claim 4b,4c are similar to Claim 4a (each illustrated in Figure 9(a) and Figure 9(b)), despite that we will always have $|(\boldsymbol{A}[l\colon l]\text{ of }G)|=1$.

11.3 The construction of a mathematical equivalent G_2 based on G_1, the definition of ($ES1_{sub}$ of G_2), and the proof of Claim 5

Claim 5. *If* $([ES1_{sub}]_S^D[L\colon L]\ of\ G)=(ES2\ of\ G)$ *, then* $(\forall\langle w,D,L\rangle \in ([ES1_{sub}]_S^D\ of\ G))\ (\exists\ \sigma-\text{path}\ S-\cdots-w-D\subseteq (ES1_{sub}\ of\ G))$.

Based on the following Step I, II and III, Claim 5 can get proved.

11.3.1 Step I: The construction of mathematical equivalent G_2 based on G_1

Our inductive proving framework requires that, we need to find some proper ($ES1_{sub}$ of G_1) which is mathematically equivalent to the provided ($ES1_{sub}$ of G). To achieve that, a graph G_2 is further constructed based on G_1. In G_2, with the help of Claim 4, we precisely list out each σ-path in G in a flavor of "mathematical analysis", as follows.

By Claim 4 and criterion (ii), for each $\langle w,D,L\rangle \in E$, if $e\in(\lambda_{sub}(w)\text{ of }G)$ ($e=\langle a,b,h\rangle\in E$, $1<h\le l$), there should exist at least one σ-path that traverses both e and $\langle w,D,L\rangle$ in G. Just pick one such σ-path for each pair of e and $\langle w,D,L\rangle$. If the σ-path traverses $\langle u_i,v,l\rangle$ for some $i\in\{1,2\}$, then we can denote it by $P^{\langle e|i|w\rangle}$; otherwise, just denote it by $P^{\langle e|0|w\rangle}$.

Then, for each such $P^{\langle e|i|w\rangle}$ (where $e=\langle a,b,h\rangle, 1<h\le l, i\in\{0,1,2\}$) in G, we introduce a new path $x_{L-2}^{(e|i|w)}-x_{L-1}^{(e|i|w)}-D$ to G_1, such that there will exist a path $X^{\langle e|i|w\rangle}$ as follows in the resulted graph G_2:

(i) For $i\in\{1,2\}$: let $X^{\langle e|i|w\rangle}=_{def}\left(u_i-v_i-x_{l+1}^{(e|i|w)}-\cdots-x_{L-2}^{(e|i|w)}\right)\cup\left(x_{L-2}^{(e|i|w)}-x_{L-1}^{(e|i|w)}-D\right)$, where $u_i-v-\cdots-w-D\subseteq P^{\langle e|i|w\rangle}\subseteq E$ and $u_i-v_i-x_{l+1}^{(e|i|w)}-\cdots-x_{L-2}^{(e|i|w)}\subseteq u_i-v_i-\cdots-w_i-D\subseteq E_1$.

(ii) For $i=0$: let $X^{\langle e|i|w\rangle}=_{def}\left(\hat{x}_{l-1}-\hat{x}_l-x_{l+1}^{(e|i|w)}-\cdots-x_{L-2}^{(e|i|w)}\right)\cup\left(x_{L-2}^{(e|i|w)}-x_{L-1}^{(e|i|w)}-D\right)$, where $\hat{x}_{l-1}-\hat{x}_l-x_{l+1}^{(e|i|w)}-\cdots-x_{L-2}^{(e|i|w)}\subseteq P^{\langle e|i|w\rangle}\subseteq E\cap E_1$.

Set λ_2 for G_2:

$$\begin{aligned}\left(\lambda\left(x_{L-1}^{(e|i|w)}\right)\ \text{of}\ G_2\right)&=P^{\langle e|i|w\rangle}[1\colon l-1]\cup X^{\langle e|i|w\rangle}[l\colon L-1]\ (\text{where } X^{\langle e|i|w\rangle}\subseteq E_2),\\ (\lambda(D)\ \text{of}\ G_2)&=(\lambda(D)\ \text{of}\ G_1)\cup\bigcup\nolimits_{X^{\langle e|i|w\rangle}\subseteq E_2}\left(x_{L-2}^{(e|i|w)}-x_{L-1}^{(e|i|w)}-D\right).\end{aligned}\tag{17}$$

All the other labels stay the same as they were defined in G_1. We thus get a new graph $G_2=<$

$\widehat{V_2}$[16], $E_2, S, D, L, \lambda_2 >$ (see Figure 6(c)).

The above "singleton" definition of $\left(\lambda\left(x_{L-1}^{\langle e|i|w\rangle}\right) \text{ of } G_2\right)$ $(X^{\langle e|i|w\rangle} \subseteq E_2)$ makes the label be controlled to correspond to exactly one σ-path $P^{\langle e|i|w\rangle}[1\colon l-1] \cup X^{\langle e|i|w\rangle}$ in G_2.

For latter usage, as with each $\hat{x}_l$ at stage l of G_2, we arbitrarily pick one path $X^{\langle e|i|w\rangle} \subseteq E_2$ (if exists, where $\hat{x}_l$ appears on $X^{\langle e|i|w\rangle}$, $i \in \{0,1,2\}$), and add one path "$x_{L-2}^{\langle e|i|w\rangle} - \omega_{L-1}^{\hat{x}_l} - D$" to G_2, such that there will exist one path $Pspare_{\hat{x}_l} =_{def} \left(\hat{x}_l - \omega_{l+1}^{\hat{x}_l} - \cdots - \omega_{L-1}^{\hat{x}_l} - D\right) = X^{\langle e|i|w\rangle}[l+1\colon L-2] \cup \left(x_{L-2}^{\langle e|i|w\rangle} - \omega_{L-1}^{\hat{x}_l} - D\right) = \left(\hat{x}_l - x_{l+1}^{\langle e|i|w\rangle} - \cdots - x_{L-2}^{\langle e|i|w\rangle}\right) \cup \left(x_{L-2}^{\langle e|i|w\rangle} - \omega_{L-1}^{\hat{x}_l} - D\right)$. Set the labels as:

$$\begin{aligned} \left(\lambda(\omega_{L-1}^{\hat{x}_l}) \text{ of } G_2\right) &= \left(P^{\langle e|i|w\rangle}[1\colon l-1] \cup X^{\langle e|i|w\rangle}[l\colon l] \cup Pspare_{\hat{x}_l}[l+1\colon L-1]\right), \\ (\lambda(D) \text{ of } G_2) &= (\lambda(D) \text{ of } G_2) \cup Pspare_{\hat{x}_l}. \end{aligned} \tag{18}$$

Note that $P^{\langle e|i|w\rangle}[1\colon l-1] \cup X^{\langle e|i|w\rangle}[l\colon l] \cup Pspare_{\hat{x}_l}$ is a σ-path, which is also exactly contained by $\left(\lambda(\omega_{L-1}^{\hat{x}_l}) \text{ of } G_2\right)$.

The number of the above newly introduced paths for all possible combinations of e,w and i is a polynomial in $|E|$.

We thus finished the construction of G_2.

It can be easily obtained again that:

Remark 1 (Claim 1 for G_2). $f(G_2) < f(G)$.

Remark 2 (Claim 2 for G_2). *If* $\left(\chi^D_{R(E)}(\lambda(D)) \text{ of } G\right) \neq \emptyset$, *we have:*

$$(ES1 \text{ of } G_2) \supseteq \left(\begin{array}{c} \left((ES1 \text{ of } G) - \{e | e \in u_i - v - \cdots - D \subseteq E, i \in \{1,2\}\}\right) \\ \cup \left\{e \middle| \begin{array}{c} e \in u_i - v_i - \cdots - w_i - D \subseteq E_1, \\ (R(\langle u_i, v, l\rangle) \text{ of } G) \neq \emptyset, i \in \{1,2\} \end{array}\right\} \\ \cup \left\{\hat{e} \middle| \begin{array}{c} \hat{e} \in X^{\langle e|i|w\rangle} \subseteq E_2, \\ i \in \{0,1,2\} \end{array}\right\} \cup \{\hat{e} | \hat{e} \in Pspare_{\hat{x}_l} \subseteq E_2\} \end{array}\right) \neq \emptyset. \tag{19}$$

And hence, $\left(\chi^D_{R(E)}(\lambda(D)) \text{ of } G\right) \neq \emptyset$ *implies*

$$(ES1 \text{ of } G_2) = \left(\chi^D_{R(E)}(\lambda(D)) \text{ of } G_2\right) \neq \emptyset. \tag{20}$$

Proof. This is clear. The renaming rules for $(R(E) \text{ of } G_2)$ is analogous to those for $(R(E) \text{ of } G_1)$, since G_2 only adds some ω-paths to G_1. Moreover, each of those "$X^{\langle e|i|w\rangle}$" is on some σ-path in G_2 and hence can be kept in $\left(\chi^D_{R(E)}(\lambda(D)) \text{ of } G_2\right)$. □

Remark 3 (Claim 3 for G_2). G_2 *satisfies Definition 3(b).*

Proof. Definition 3(b) is clearly fulfilled, because: (i) we didn't change the in-degrees of other vertices except D in G_1; (ii) the vertices on those newly introduced "$X^{\langle e|i|w\rangle}[l+1\colon L-1]$" are single-in-degree vertices. □

[16] Note again that $\widehat{V_n}$ is used to distinguish from the notation "V_n" in Definition 1.

11.3.2 Step III: The definition of $(ES1_{sub}$ of $G_2)$

11.3.2.1 Step III(a): The initial definition of $(ES1_{sub}$ of $G_2)$

Guided by the given $(ES2$ of $G)$, $(\lambda_{sub}(w)$ of $G)$ and $(ES1_{sub}$ of $G)$, we can define an initial $(ES1_{sub}$ of $G_2)$ as follows:

- $(ES2 \text{ of } G_2) =_{\text{def}} \left\{\hat{e} \middle| \hat{e} \in X^{\langle e|i|w\rangle}[L\colon L] \subseteq E_2,\ \langle w, D, L\rangle \in (ES2 \text{ of } G)\right\}$.
- $(ES1_{sub}[l+1\colon L] \text{ of } G_2) =_{\text{def}} \left\{\hat{e} \middle| \hat{e} \in X^{\langle e|i|w\rangle}[l+1\colon L],\ \langle x_{L-1}^{(e|i|w)}, D, L\rangle \in (ES2 \text{ of } G_2)\right\}$.

(Substitute "$P^{\langle e|i|w\rangle}[l+1\colon L]$" by "$X^{\langle e|i|w\rangle}[l+1\colon L]$" in G_2, if $\langle w, D, L\rangle \in (ES1_{sub}$ of $G)$.)

- $(ES1_{sub}[1\colon l] \text{ of } G_2) =_{\text{def}} ([ES1_{sub}]_S^D[1\colon l] \text{ of } G)$[17].
- $\left(\lambda_{sub}\left(x_{L-1}^{(e|i|w)}\right) \text{ of } G_2\right) =_{\text{def}}$

$$\begin{cases} \begin{pmatrix} ([ES1_{sub}]_S^D \text{ of } G) \cap \\ \left(\lambda\left(x_{L-1}^{(e|i|w)}\right)[1\colon L-2] \text{ of } G_2\right) \end{pmatrix} \cup \left\{\langle x_{L-2}^{(e|i|w)}, x_{L-1}^{(e|i|w)}, L-1\rangle\right\}, & i=0 \\ \begin{pmatrix} ([ES1_{sub}]_S^D \text{ of } G)^{18} \cap \\ \left(\lambda\left(x_{L-1}^{(e|i|w)}\right)[1\colon l] \text{ of } G_2\right) \end{pmatrix} \cup X^{\langle e|i|w\rangle}[l+1\colon L-1], & i \in \{1,2\} \end{cases}$$

where $\langle x_{L-1}^{(e|i|w)}, D, L\rangle \in (ES2 \text{ of } G_2)$.

(Partition the set "$([ES1_{sub}]_S^D$ of $G)$" by "$\left(\lambda\left(x_{L-1}^{(e|i|w)}\right)$ of $G_2\right)$".)

The motivation of the construction is as follows.

If $(ES1_{sub}$ of $G)$ contains $\langle w, D, L\rangle \in E$ (hence there exists $P^{\langle e|i|w\rangle}$ in G by Claim 4, where $e \in E$ and $i \in \{0,1,2\}$), we use all related "$X^{\langle e|i|w\rangle}[l+1\colon L]$" in G_2 to substitute "$P^{\langle e|i|w\rangle}[l+1\colon L]$" (note that $P^{\langle e|i|w\rangle}[l+1\colon L] \subseteq (ES1_{sub}$ of $G)$, by $([ES1_{sub}]_S^D[L\colon L]$ of $G) = (ES2$ of $G)$ and by the structure of G). Further, we can exactly choose edges for $\left(\bigcup_{\langle y,D,L\rangle \in ES2} \lambda_{sub}(y) \text{ of } G_2\right)[1\colon l]$, such that $\left(\bigcup_{\langle y,D,L\rangle \in ES2} \lambda_{sub}(y) \text{ of } G_2\right)[1\colon l]$ is essentially the same as $\left(\bigcup_{\langle w,D,L\rangle \in ES2} \lambda_{sub}(w) \text{ of } G\right)[1\colon l]$. Consequently, different from the computation of $([ES1_{sub}]_S^D$ of $G)$, the computation of $([ES1_{sub}]_S^D$ of $G_2)$ will use all those related "$X^{\langle e|i|w\rangle}[l+1\colon L]$" instead of "$P^{\langle e|i|w\rangle}[l+1\colon L]$".

To keep consistent with the criteria on the constitution of $(ES1_{sub}$ of $G_2)$, the set $\left(\lambda_{sub}\left(x_{L-1}^{(e|i|w)}\right) \text{ of } G_2\right)$ $(\langle x_{L-1}^{(e|i|w)}, D, L\rangle \in (ES2 \text{ of } G_2))$ is accordingly defined, by partitioning the set $([ES1_{sub}]_S^D$ of $G)$ using each σ-path "$P^{\langle e|i|w\rangle}[1\colon l-1] \cup X^{\langle e|i|w\rangle}[l\colon L]$" specified by "$\left(\lambda\left(x_{L-1}^{(e|i|w)}\right) \text{ of } G_2\right)$".

The benefit of this construction lies in that, as will be certificated later, the usage of "$X^{\langle e|i|w\rangle}[l+1\colon L]$" and the "singleton" definition of "$\left(\lambda\left(x_{L-1}^{(e|i|w)}\right) \text{ of } G_2\right)$" will make it easier and clearer to "recover" those σ-paths "$P^{\langle e|i|w\rangle}$"$\subseteq (ES1_{sub}$ of $G)$ from those σ-paths "$P^{\langle e|i|w\rangle}[1\colon l-$

---

[17] Mild abuse of notation for readability. To be exact, write $\bigcup_{e \in ([ES1_{sub}]_S^D[1:l] \text{ of } G)} \left\{\hat{e} \middle| \begin{matrix} \hat{e} = \langle u_i, v_i, l\rangle \text{ if } e = \langle u_i, v, l\rangle; \\ \hat{e} = e \text{ otherwise. } (i \in \{1,2\}) \end{matrix}\right\}$.

[18] Similar mild abuse of notation to the definition of $(ES1_{sub}[1\colon l]$ of $G_2)$.

1] $\cup X^{\langle e|i|w\rangle}$"$\subseteq (ES1_{sub}$ of $G_2)$.

Apparently, $(ES1_{sub}$ of $G_2) = \left(\left(ES2 \cup \left(\left(\bigcup_{\langle y,D,L\rangle \in ES2} \lambda_{sub}(y)\right) \cap ES1\right)\right)$ of $G_2\right)$ and $(\lambda_{sub}(y)$ of $G_2) \subseteq ((\lambda(y)[1:1] \cup \{e \in \lambda(y)[2:L] | \langle y, D, L\rangle \in R(e)\})$ of $G_2)$ $(\langle y, D, L\rangle \in E_2)$. If v does not appear on some $S - \cdots - a_i - \cdots - D \subseteq ([ES1_{sub}]_S^D$ of $G)$ or it just appears with $i < l$, then there exists $S - a_1 - \cdots - a_i \subseteq ([ES1_{sub}]_S^D$ of $G_2)$ such that we still have that "$([ES1_{sub}]_S^D$ of $G_2)$ contains one $\langle a_j, *, j+1\rangle$ at most for each a_j $(1 \le j < i)$ while two $\langle a_i, *, i+1\rangle$ at least, and $S - a_1 - \cdots - a_i \not\subseteq (\lambda_{sub}(y)$ of $G_2)$ for $\langle y, D, L\rangle \in (ES2$ of $G_2)$". If $v = a_i$ and hence appears on some $S - \cdots - a_i - \cdots - D \subseteq ([ES1_{sub}]_S^D$ of $G)$, then there exists some $P = S - \cdots - v_1 - \cdots - t_1 \subseteq E_2$ (where t_1 lies at stage $L-2$) and $P[1:l] \not\subseteq (\lambda_{sub}(y)$ of $G_2)$ for $\langle y, D, L\rangle \in (ES2$ of $G_2)$. Therefore we have:

Remark 4 (Initial $(ES1_{sub}$ of $G_2)$, on the constitution). *The initially defined $(ES1_{sub}$ of $G_2)$ and $\left(\lambda_{sub}\left(x_{L-1}^{(e|i|w)}\right)$ of $G_2\right)$ $(\langle x_{L-1}^{(e|i|w)}, D, L\rangle \in (ES2$ of $G_2))$ obeys the criteria (i),(ii),(iii),(iv),(v) by the PA.*

11.3.2.2 Step III(b): The homomorphic compensation to $(ES1_{sub}$ of $G_2)$ to prove Claim 5

For the initially defined $(ES1_{sub}$ of $G_2)$, a trouble might be aroused around the connectivity prerequisite $([ES1_{sub}]_S^D[L:L]$ of $G_2) = (ES2$ of $G_2)$, due to the "split" of the multi-in-degree vertex v. Further "**compensation**" to $(ES1_{sub}$ of $G_2)$ should be done conditionally, to keep $(ES1_{sub}$ of $G_2)$ mathematically equivalent to the provided $(ES1_{sub}$ of $G)$ and to maintain the connectivity prerequisite. The query [19] of such kind $(ES1_{sub}$ of $G_2)$ is the deepest and core discussion of the whole paper.

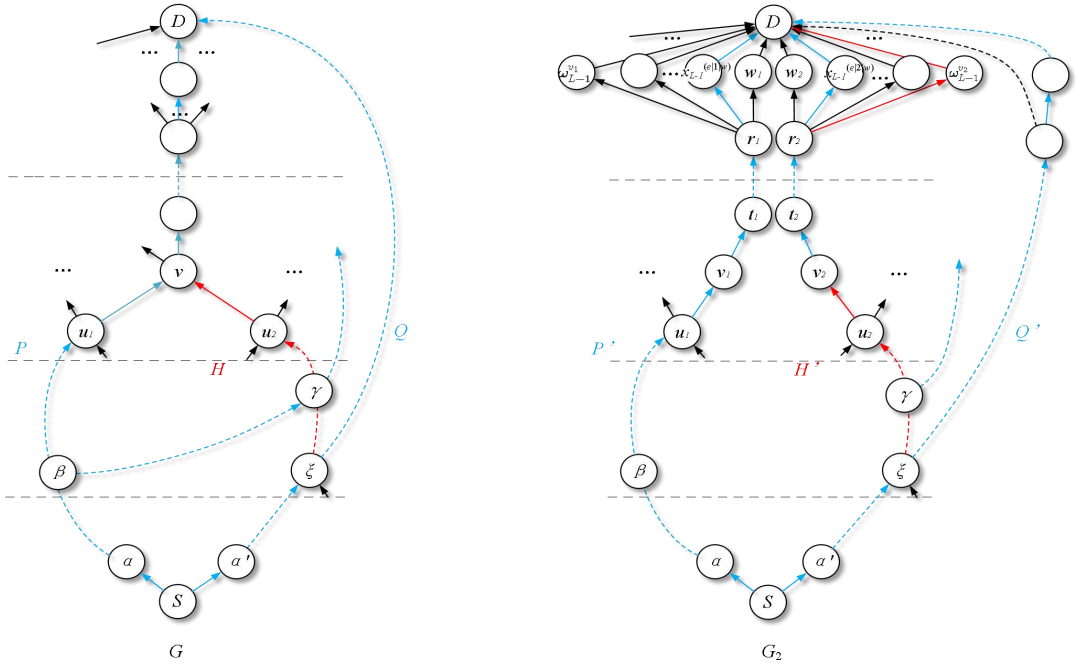


Figure 10: A typical case of the homomorphic compensation

[19] This method (i.e., the existence of this homomorphic compensation, which might have not ever been established by all previous studies), we think, just has the same logical power with the seek and construction of uncomputable functions [Chu36, Tur36] by the method of diagonalization.

Case 1. $\left\{\begin{matrix}\langle u_1,v,l\rangle,\\ \langle u_2,v,l\rangle\end{matrix}\right\}\cap([ES1_{sub}]^D_S \text{ of } G)=\left\{\langle u_1,v,l\rangle\middle|\left((ES2\cap R(\langle u_2,v,l\rangle)) \text{ of } G\right)\neq\emptyset\right\}$ [20].

The homomorphic compensation is done constructively, as follows. The idea is to find some proper path to repair the connectivity prerequisite if $\langle u_2,v,l\rangle$ does not appear in $(ES1_{sub}$ of $G_2)$.

Case 1a. $\left|([ES1_{sub}]^D_S[L\colon L] \text{ of } G)\right|>1$.

In this case, there exists $S-a_1-\cdots-a_i\subseteq([ES1_{sub}]^D_S$ of $G)$ $(i>1)$ such that $([ES1_{sub}]^D_S$ of $G)$ contains one $\langle a_j,*,j+1\rangle$ at most for each a_j $(1\leq j<i)$ while two $\langle a_i,*,i+1\rangle$ at least, and $S-a_1-\cdots-a_i\nsubseteq(\lambda_{sub}(y)$ of $G)$ for $\langle y,D,L\rangle\in(ES2$ of $G)$.

Pick a path $P\subseteq(ES1_{sub}$ of $G)$ that contains $S-a_1-\cdots-a_i$; correspondingly, there exists $P'\subseteq(ES1_{sub}$ of $G_2)$ that contains $S-a_1-\cdots-a_i$ too (if $a_j=v$ in G, then denote $a_j=v_1$ in G_2, where $1\leq j\leq i$). In G, from $\langle u_2,v,l\rangle$ down to S, we seek for such a path: every time the path will reach to meet P, we choose another edge to continue our "downward searching" (we have $d^-(x)>1$ for each path $x-\cdots-v\subseteq E$ that starts above stage 1 by item 3 of Definition 3(b)). Hence, search down until we meet any path $Q\subseteq([ES1_{sub}]^D_S$ of $G)$ of a "branch" other than the one containing P (hence $Q\cap\{\langle u_1,v,l\rangle,\langle u_2,v,l\rangle\}=\emptyset$), before reaching S; otherwise, we directly reach S and just regard $Q=\emptyset$ might as well.

Let the vertex of intersection be ξ and the traversed path be $H=\xi-\cdots-v$, as illustrated in Figure 10. Turn $T=[Q]^\xi_S\cup H[1\colon l-1]\cup\{\langle u_2,v_2,l\rangle\}\cup Pspare_{v_2}$ into a σ-path of G_2, by properly **expanding** the labels on T. **Add** $[T]^{v_2}_\xi$ to $(ES1_{sub}$ of $G_2)$, by setting $(\lambda(\omega^{v_2}_{L-1})$ of $G_2)=_{\text{def}}T[1\colon L-1]$, $(\lambda_{sub}(\omega^{v_2}_{L-1})$ of $G_2)=_{\text{def}}T[1\colon L-1]$ and $(ES2$ of $G_2)=_{\text{def}}(ES2$ of $G_2)\cup\{\langle\omega^{v_2}_{L-1},D,L\rangle\}$. If H intersects with other "subbranches" of the branch containing P—i.e., there exists some "bridge" $\beta-\cdots-\gamma\subseteq([ES1_{sub}]^D_S$ of $G)$ such that $\beta\in\bigcup_{0<i<l-1}V_i$ on P and $\gamma\in\bigcup_{1<i<l}V_i$ on H, **prune** the entire "bridge" from each $(\lambda_{sub}(y)$ of $G_2)$ $(\langle y,D,L\rangle\in(ES2$ of $G_2))$, so that $(ES1_{sub}$ of $G_2)$ obeys criteria (iv).

Now, with the help of $[T]^{v_2}_\xi$, we have $([ES1_{sub}]^D_S[L\colon L]$ of $G_2)=(ES2$ of $G_2)$.— Each $X^{\langle e|2|w\rangle}[l+1\colon L]$ becomes a "subbranch" of T and some "subbranches" of the branch that contains P become "subbranches" of T. Hence the pruning of "bridges" won't disturb the connectivity. We already expanded $(ES2$ of $G_2)$ to include $\langle\omega^{v_2}_{L-1},D,L\rangle$ and we also expanded $(ES1_{sub}$ of $G_2)$ to include T ($[T]^\xi_S$ is originally in $(ES1_{sub}$ of $G_2)$).

Note that we did not change the criteria (i),(ii) which $([ES1_{sub}]^D_S$ of $G_2)$ should obey. Criterion (iv) is obeyed since all possible "bridges" are pruned. Since each $\left(\lambda_{sub}\left(x^{(e|i|w)}_{L-1}\right)\text{ of }G_2\right)$ is defined to exactly contain one σ-path, expanding labels on T bring us no more σ-paths than T. We have set $(\lambda(\omega^{v_2}_{L-1})$ of $G_2)=T[1\colon L-1]$ and $(\lambda_{sub}(\omega^{v_2}_{L-1})$ of $G_2)=T[1\colon L-1]$, hence T is a unique σ-path that is newly increased. Therefore, $(ES1_{sub}$ of $G_2)$ obeys criterion (iii).

The discussion on the obedience of criterion (v) by the initial $(ES1_{sub}$ of $G_2)$ still holds after compensation: there will exist some $S-a_1-\cdots-a_{\hat{i}}\subseteq([ES1_{sub}]^D_S$ of $G_2)$ $(1<i\leq\hat{i}$ due to the "split" of $v)$ such that we still have that $([ES1_{sub}]^D_S$ of $G_2)$ contains one $\langle a_j,*,j+1\rangle$ at most for each a_j $(1\leq j<\hat{i})$ while two $\langle a_{\hat{i}},*,\hat{i}+1\rangle$ at least, and $S-a_1-\cdots-a_{\hat{i}}\nsubseteq(\lambda_{sub}(y)$ of $G_2)$ for $\langle y,D,L\rangle\in(ES2$ of $G_2)$, because we did not change any "$\left(\lambda_{sub}\left(x^{(e|i|w)}_{L-1}\right)\text{ of }G_2\right)$" and $(\lambda_{sub}(\omega^{v_2}_{L-1})$ of $G_2)$ only contains $T[1\colon L-1]$. Meanwhile, ξ is never on $S-a_1-\cdots-a_i$.

Then a contradiction will arise. According to (H1), we have $(\forall\langle y,D,L\rangle\in$

[20] The discussion is symmetrical if choosing $\langle u_2,v,l\rangle$.

$([ES1_{sub}]^D_S \text{ of } G_2)\big)\,(\exists\,\sigma-\text{path } S-\cdots-y-D\subseteq(ES1_{sub} \text{ of } G_2))$ (note that y is some "$x^{(e|i|w)}_{L-1}$"). Those σ-paths can't be T, since $(\lambda_{sub}(\omega^{v_2}_{L-1}) \text{ of } G_2)$ only contains $T[1\colon L-1]$. Then it can be inferred that there exists a σ-path $SP=S-a_1-\cdots-a_i-\cdots-w-D\subseteq(ES1_{sub} \text{ of } G)$. Since $(ES1_{sub} \text{ of } G)$ obeys criteria (i),(ii),(iii),(iv),(v), $S-a_1-\cdots-a_i\subseteq SP[1\colon L-1]\subseteq(\lambda_{sub}(w) \text{ of } G)$. Criterion (v) is violated.

Case 1b. $\big|([ES1_{sub}]^D_S[L\colon L] \text{ of } G)\big|=1$.

If the unique path $P=S-\cdots-D\subseteq([ES1_{sub}]^D_S \text{of } G)$ is a σ-path, Claim 5 is just proved. If P is not a σ-path, after splitting, we must have no less than two of those "$\langle x^{(e|i|w)}_{L-1},D,L\rangle$" ($i\in\{1,2\}$) in $(ES2 \text{ of } G_2)$. All the related "$\left(\lambda_{sub}\left(x^{(e|i|w)}_{L-1}\right) \text{ of } G_2\right)$" contain $P'[1\colon L-2]$ (where $P'=P[1\colon l-1]\cup(u_1-v_1-\cdots-D)$) when united together, but no one individually contains the whole $P'[1\colon L-2]$. Then the subsequent discussion can totally shift to the above discussion of Case 1a. We will encounter a contradiction again.

Summarizing the above discussions of Case 1a and Case 1b, $([ES1_{sub}]^D_S \text{ of } G)$ can only contain a unique path and the path must be a σ-path. Hence $(\forall\langle w,D,L\rangle\in([ES1_{sub}]^D_S \text{ of } G))\,(\exists\,\sigma-\text{path } S-\cdots-w-D\subseteq(ES1_{sub} \text{ of } G))$.

Case 2. $\left\{\begin{matrix}\langle u_1,v,l\rangle,\\ \langle u_2,v,l\rangle\end{matrix}\right\}\cap([ES1_{sub}]^D_S \text{ of } G)\in\left\{\begin{matrix}\emptyset,\\ \left\{\langle u_1,v,l\rangle\middle|\left((ES2\cap R(\langle u_2,v,l\rangle)) \text{ of } G\right)=\emptyset\right\}\end{matrix}\right\}$.

Note that $([ES1_{sub}]^D_S \text{ of } G)$ cannot contain both in-degrees of v by criterion (iv), and the condition $\left((ES2\cap R(\langle u_2,v,l\rangle)) \text{ of } G\right)=\emptyset$ implies that there will exist no $\langle x^{(e|2|w)}_{L-1},D,L\rangle\in(ES2 \text{ of } G_2)$.

In Case 2, $([ES1_{sub}]^D_S[L\colon L] \text{ of } G_2)=(ES2 \text{ of } G_2)$ already and no compensation is needed. Note that $([ES1_{sub}]^D_S \text{ of } G_2)$ now keeps naturally the same with $([ES1_{sub}]^D_S \text{ of } G)$ —for $\langle w,D,L\rangle\in([ES1_{sub}]^D_S \text{ of } G)$, there always exists some $\langle x^{(e|i|w)}_{L-1},D,L\rangle\in([ES1_{sub}]^D_S \text{ of } G_2)$. Since $([ES1_{sub}]^D_S[L\colon L] \text{ of } G_2)=(ES2 \text{ of } G_2)$, $(ES1_{sub} \text{ of } G_2)$ obeys criteria (i),(ii),(iii),(iv),(v) (Remark 4) and $f(G_2)<f(G)$, we know that $\left(\forall\langle y,D,L\rangle\in([ES1_{sub}]^D_S \text{ of } G_2)\right)\,(\exists\,\sigma-\text{path } S-\cdots-y-D\subseteq(ES1_{sub} \text{ of } G_2))$ by (H1). It follows that $\left(\forall\langle w,D,L\rangle\in([ES1_{sub}]^D_S \text{ of } G)\right)\,(\exists\,\sigma-\text{path } S-\cdots-w-D\subseteq(ES1_{sub} \text{ of } G))$.

We thus finish the proof of Claim 5.

The above Claim 1,2,3,4,5 conclude the proof of Lemma 2.

12 Proof of Theorem 6

Theorem 6. *If the compact kernel of G is not empty, there exists a σ-path in G.*

Proof. Let $G=<V,E,S,D,L,\lambda>$ be the multi-stage graph in the $2-$MSP inputted to the ZH algorithm.

Pick any $\langle y,D,L\rangle\in ES1[L\colon L]$, we can choose $ES2=\{\langle y,D,L\rangle\}$. Since $[R(\langle x,y,L-1\rangle)\cap ES1]^D_y\neq\emptyset$ for an arbitrary $\langle x,y,L-1\rangle\in ES1$, then $\langle y,D,L\rangle\in R(\langle x,y,L-1\rangle)$ and there exists $P=S-a-\cdots-y-D\subseteq A$ for the set A computed when deciding $\langle y,D,L\rangle\in R(\langle x,y,L-1\rangle)$,

we can choose $\lambda_{sub}(y) = P[1{:}\,L-1] \subseteq \lambda(y)[1{:}\,1] \cup \{e \in \lambda(y)[2{:}\,L] | \langle y, D, L\rangle \in R(e)\}$.

Then, we can obtain $ES1_{sub} = ES2 \cup (\lambda_{sub}(y) \cap ES1) \subseteq ES1$.

Further, we can obtain that $[ES1_{sub}]_S^D[L{:}\,L] = ES2 \neq \emptyset$, when noting that $A \subseteq ES1$.

$ES1_{sub}$ fulfills the criteria (i),(ii),(iv). $ES1_{sub}$ cannot obey criterion (iii), because we can assume G has no σ-paths. $ES1_{sub}$ obeys criterion (v), since $|ES2| = 1$.

Then, G must contain a σ-path $SP \subseteq ES1_{sub}$ as claimed by step 3 of the PA by the $\alpha\beta$ lemma. □

13 Supplementary materials for the proof of Lemma 2

13.1 The renaming rules and the "transit" technique for $(R(E)$ of $G_1)$

For every $\{\langle r, s, k\rangle, \langle o, p, h\rangle\} \subseteq E$ ($1 \leq k < h \leq L$), if $\langle o, p, h\rangle \in (R(\langle r, s, k\rangle)$ of $G)$ (where $(R(\langle r, s, k\rangle)$ of $G) \in (R(E)$ of $G)$), there must exist $\{e_1, e_2\} \subseteq E_1$, such that $e_2 \in (R(e_1)$ of $G_1)$ (where $(R(e_1)$ of $G_1) \in (R(E)$ of $G_1)$). This should hold for both initial and constrained ρ-path edge-sets. Here are the detailed renaming rules for the above e_1,e_2.

The renaming rules for $(R(E)$ of $G_1)$:

Case 1. $(\langle o, p, h\rangle \notin ((u_i - v - \cdots - D)$ of $G)$, $i \in \{1,2\})$:
$e_1 = \langle r, s, k\rangle$, $e_2 = \langle o, p, h\rangle$.

Case 2. $(\langle o, p, h\rangle \in ((u_i - v - \cdots - D)$ of $G)$, $i \in \{1,2\})$:

2.1. $\langle o, p, h\rangle = (\langle u_i, v, l\rangle$ of $G)$:
$e_1 = \langle r, s, k\rangle$, $e_2 = \langle u_i, v_i, l\rangle$.

2.2. $(\langle o, p, h\rangle \in ((v - \cdots - D)$ of $G))$:

2.2.1. $k < l$:
$e_1 = \langle r, s, k\rangle$, $e_2 \in \big((v_j - \cdots - D)[h{:}\,h]$ of $G_1\big)$
(such that $\langle u_j, v, l\rangle \in (R(\langle r, s, k\rangle)$ of $G)$[21], $j \in \{1,2\}$).

2.2.2. $k = l$:
$e_1 = \langle u_j, v_j, l\rangle$, $e_2 \in \big((v_j - \cdots - D)[h{:}\,h]$ of $G_1\big)$
(such that $\langle r, s, k\rangle = \langle u_j, v, l\rangle$, $j \in \{1,2\}$).

2.2.3. $k > l$:
$e_1 \in \big((v_j - \cdots - D)[k{:}\,k]$ of $G_1\big)$, $e_2 \in \big((v_j - \cdots - D)[h{:}\,h]$ of $G_1\big)$
(such that $\langle o, p, h\rangle \in (R(\langle u_j, v, l\rangle)$ of $G)$, $j \in \{1,2\}$).

The key technique used here—to clarify the discussion of the many edges and paths involved in the computations of Operator 2,3,4—is a **"transit"** between $(R(e)$ of $G)$ and $(R(e)$ of $G_1)$ on the multi-in-degree vertex v in G:

(1) Once $\langle u_i, v, l\rangle \in (R(e)$ of $G)$ ($i \in \{1,2\}$), then $\langle u_i, v_i, l\rangle \in (R(e)$ of $G_1)$ (by the radical expansion on $(\lambda(v_i)$ of $G_1)$);

---

[21] If $\{\langle u_1, v, l\rangle, \langle u_2, v, l\rangle\} \subseteq (R(\langle r, s, k\rangle)$ of $G)$, there are two edges (i.e., the ones in $\bigcup_{j\in\{1,2\}} \big((v_j - \cdots - D)[h{:}\,h]\big)$) in G_1 each corresponding to the $\langle o, p, h\rangle$ in G. It's similar for the other cases.

(2) Then straightly, we can have $v_i - \cdots - w_i - D \subseteq (R(e) \text{ of } G_1)$ (by the radical expansion on $(\lambda(t_i) \text{ of } G_1)$,...,$(\lambda(w_i) \text{ of } G_1)$).

The "transit" is the direct consequence of the radical expansion. It ensures that, the initial ρ-paths (by Operator 2) and constrained ρ-paths (by Operator 4) of each edge in G are "naturally preserved" in G_1 despite the "split" of v, when computing $(R_0(e) \text{ of } G_1)$ by the definition of Operator 2 and when computing $(R(e) \text{ of } G_1)$ by the definition of Operator 4. It saves the heavy efforts otherwise required to dive into the details of the operators, especially the convoluted Operator 4.

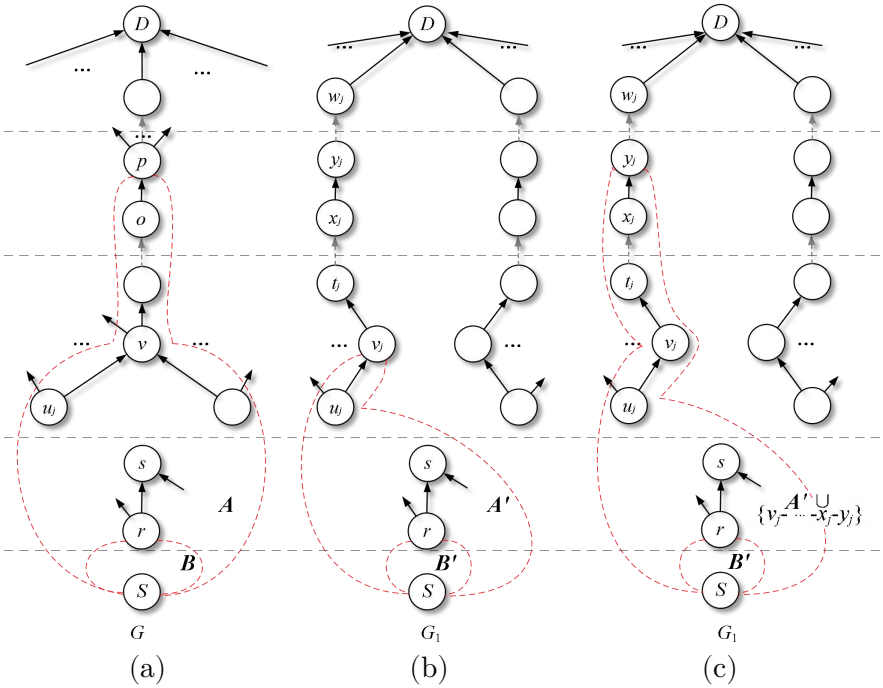


Figure 11: A typical case (Case 2.2.1) of the renaming rules for $(R(E) \text{ of } G_1)$

Now, let's utilize this technique to explain the above naming rules. [22] For $\langle o, p, h\rangle \in (R(\langle r, s, k\rangle) \text{ of } G)$, w.l.o.g., let's suppose $k < l < h$ (i.e., Case 1 or Case 2.2.1, other cases are analogous):

(i) When $k \leq 2$, things become trivial, since $(R(\langle r, s, k\rangle) \text{ of } G_1) = (R_0(\langle r, s, k\rangle) \text{ of } G_1)$ (recall the ZH algorithm skips the first two stages). Thus, assume $k > 2$ hereinafter.

(ii) If $\langle o, p, h\rangle \notin [E]_v^D$, then $\{\langle r, s, k\rangle, \langle o, p, h\rangle\} \subseteq E_1$. By the construction of G_1, $\langle o, p, h\rangle \in (R(\langle r, s, k\rangle) \text{ of } G_1)$.

(iii) Otherwise if $\langle o, p, h\rangle \in [E]_v^D$, by the definition of Operator 4, the set $(\boldsymbol{A} \text{ of } G)$ involved in the computation for deciding "$\langle o, p, h\rangle \in (R(\langle r, s, k\rangle) \text{ of } G)$" is non-empty (see Figure 11(a)). Since $\{\langle u_i, v, l\rangle, \langle o, p, h\rangle\} \subseteq (\boldsymbol{A} \text{ of } G)$ $(i \in I, I \subseteq \{1,2\})$, then by the definition of Operator 3 for computing $(\boldsymbol{A} \text{ of } G)$, $\langle u_j, v, l\rangle \in (R(\langle r, s, k\rangle) \text{ of } G)$ $(j \in J, J \subseteq I)$.

(iv) By step (1) of the "transit" technique, we then have $\langle u_j, v_j, l\rangle \in (R(\langle r, s, k\rangle) \text{ of } G_1)$ (see

[22] We felt it unnecessary to exquisitely use inductive proof here, under the technique of "transit" and for the brevity of the proof; although it might will be slightly more rigorous in that manner. Besides, the ZH algorithm already naturally provides a inductive framework with the iterative steps of its pseudo-code. Throughout the iterative steps, the result to be proved is consistently guaranteed by the "transit" technique.

Figure 11(b)). By step (2) of the "transit" technique, we further have $v_j - \cdots - w_j - D \subseteq (R(\langle r,s,k\rangle)$ of $G_1)$ (see Figure 11(c)).

Thus, there exists $\langle r,s,k\rangle$ and $e_2 \in (v_j - \cdots - D)[h{:}\,h]$ in G_1, such that $e_2 \in (R(\langle r,s,k\rangle)$ of $G_1)$.

13.2 The computation of $\left(\chi^{D}_{R(E)}(\boldsymbol{ES\_temp})\right.$ of $\left.\boldsymbol{G_1}\right)$

For the chosen $(ES\_temp$ of $G_1)$, we can obtain that $\left(\chi^{D}_{R(E)}(ES\_temp)\right.$ of $\left.G_1\right) \neq \emptyset$. The argument for it is as follows.

Firstly, since $(\boldsymbol{A}$ of $G) \neq \emptyset$ and $(\boldsymbol{B}$ of $G) \neq \emptyset$, $(\boldsymbol{A}$ of $G)$ must contain $\langle u_i, v, l\rangle$ for $i \in I$ ($I \subseteq \{1,2\}$), and then we must have $u_i - v_i - \cdots - w_i - D \subseteq (ES\_temp$ of $G_1)$.

Further, we are to show that there exists a non-empty edge set $(\boldsymbol{A}'$ of $G_1)$ in G_1 computed essentially the same as the set $(\boldsymbol{A}$ of $G)$ in G, where

$$(\boldsymbol{A}'\text{ of }G_1) = (\boldsymbol{A}[1{:}\,l-1]\text{ of }G) \cup \left\{e' \middle| \begin{array}{l} e' \in u_i - v_i - \cdots - w_i - D \subseteq E_1, \\ \langle u_i, v, l\rangle \in (\boldsymbol{A}\text{ of }G), i \in \{1,2\} \end{array}\right\}. \tag{21}$$

- Firstly, by the radical expansion, $u_i - v_i - t_i - \cdots - w_i - D$ is a ω-path in G_1 and hence $([R(\langle c,d,k\rangle) \cap \boldsymbol{A}']^{D}_{d}$ of $G_1) \neq \emptyset$ for each $\langle c,d,k\rangle \in (\boldsymbol{A}'$ of $G_1)$ $(l \leq k < L)$.
- Secondly, for each $\langle c,d,k\rangle \in (\boldsymbol{A}[1{:}\,l-1]$ of $G) \subseteq (\boldsymbol{A}'$ of $G_1)$ ($k < l$), since $\langle c,d,k\rangle \in (\boldsymbol{A}$ of $G) = \left(\chi^{D}_{R(E)}(\boldsymbol{A})\right.$ of $\left.G\right)$, then each $P_1 = d - \cdots - D \subseteq ([R(\langle c,d,k\rangle) \cap \left.\chi^{D}_{R(E)}(\boldsymbol{A})\right]^{D}_{d}$ of $\left.G\right)$ must traverse $\langle u_i, v, l\rangle$ for some $i = 1$ or 2. Hence, $\langle u_i, v_i, l\rangle \in (R(\langle c,d,k\rangle)$ of $G_1)$ (by the radical expansion of $(\lambda(v_i)$ of $G_1))$ and further $u_i - v_i - t_i - \cdots - w_i - D \subseteq (R(\langle c,d,k\rangle)$ of $G_1)$ (by the radical expansion of $(\lambda(t_i)$ of $G_1)$,...,$(\lambda(w_i)$ of $G_1))$. (Also see the renaming rules and the "transit" technique discussed in Section 13.1 for $(R(E)$ of $G_1)$ if needed.) Thus, $P_2 = [P_1]^{u_i}_{d} \cup (u_i - v_i - \cdots - w_i - D) \subseteq ([R(\langle c,d,k\rangle) \cap \boldsymbol{A}']^{D}_{d}$ of $G_1)$.
- Subsequently, by the definition of Operator 3, we have $(\boldsymbol{A}'$ of $G_1) = \left(\chi^{D}_{R(E)}(\boldsymbol{A}')\right.$ of $\left.G_1\right) \neq \emptyset$.

Finally, we now intent to show that the aforementioned edge set $(\boldsymbol{A}'$ of $G_1)$ will still be non-empty when compacted by Operator 3, even all its edges at stage h are removed except $\langle a,b,h\rangle$; in other words, $\left(\chi^{D}_{R(E)}(ES\_temp)\right.$ of $\left.G_1\right) \neq \emptyset$. To show this, for each $\hat{e} = \langle c,d,k\rangle \in (ES\_temp$ of $G_1)$, consider the following cases (akin to the discussion happened during the proof of Claim 2 for G_1):

(i) $h < k < L$. Straightforwardly, $\langle c,d,k\rangle \in (ES\_temp$ of $G_1)$ implies $\langle c,d,k\rangle \in (\boldsymbol{A}'$ of $G_1)$. Hence, $([R(\langle c,d,k\rangle) \cap ES\_temp]^{D}_{d}$ of $G_1) = ([R(\langle c,d,k\rangle) \cap \boldsymbol{A}']^{D}_{d}$ of $G_1) \neq \emptyset$.

(ii) $k = h$. Then, we have $\langle c,d,k\rangle = \langle a,b,h\rangle$. Analogous to (i), we can obtain that $([R(\langle c,d,k\rangle) \cap ES\_temp]^{D}_{d}$ of $G_1) \neq \emptyset$.

(iii) $1 \leq k < h$. Since $(\boldsymbol{B}$ of $G) = \left(\chi^{a}_{R(E)}(\boldsymbol{B})\right.$ of $\left.G\right) \subseteq (ES\_temp$ of $G_1)$ is also a compacted edge set by Operator 3, it is sufficient to only consider those $\langle c,d,k\rangle \in (\boldsymbol{B}$ of $G)$ for

$\left(\chi^{D}_{R(E)}(ES\_temp) \text{ of } G_1\right)$ (see Figure 8). Since for the arbitrary enumerated $\langle c,d,k\rangle \in \left(\chi^{a}_{R(E)}(\boldsymbol{B}) \text{ of } G\right) \subseteq (ES\_temp \text{ of } G_1)$, there exists $P_1 = d - \cdots - a - b - \cdots - u_i - v - \cdots - D \subseteq ((R(\langle c,d,k\rangle) \cap \boldsymbol{A}) \text{ of } G)$ (for some $i \in \{1,2\}$), then $[P_1]_d^{u_i} \cup \langle u_i, v_i, l\rangle \subseteq (R(\langle c,d,k\rangle) \text{ of } G_1)$ (by the radical expansion on $(\lambda(v_i) \text{ of } G_1)$) and $P_2 = [P_1]_d^{u_i} \cup (u_i - v_i - t_i - \cdots - w_i - D) \subseteq ((R(\langle c,d,k\rangle) \cap ES\_temp) \text{ of } G_1)$ (by the radical expansion on $(\lambda(t_i) \text{ of } G_1)$,...,$(\lambda(w_i) \text{ of } G_1)$). (Also see the renaming rules and the "transit" technique discussed in Section 13.1 for $(R(E) \text{ of } G_1)$ if needed.)

Therefore, by the definition of Operator 3, we obtain that $\left(\chi^{D}_{R(E)}(ES\_temp) \text{ of } G_1\right) \neq \emptyset$.

14 Motivating running instances (K − SAT)

The ZH algorithm has been examined on a wide range of test cases, including random $2-$MSP instances and hard SAT instances generated using the Pigeonhole Formulas [TSZ10] and the RB model [XL00] (interestingly, it was claimed that even small instances can be hard near the phase-transition point chosen by the RB model [XBH+07]). Here, we illustrate a family of minimal unsatisfiable (abbr. MU) formulae—which was frequently mistaken by the audience as the "counter example". The formulae are:

$$\begin{aligned} F_n = & \left(\bigvee\nolimits_{1\le i\le n} x_i\right) \\ & \wedge \bigwedge\nolimits_{1\le i\le n} \left(\sim x_i \vee \bigvee\nolimits_{1\le j\le n, j\neq i} x_j\right) \\ & \wedge \bigwedge\nolimits_{1\le i<j\le n} \left((\sim x_i \vee \sim x_j) \vee \bigvee\nolimits_{1\le k\le n, k\neq i,j} x_k\right) \\ & \cdots \\ & \wedge \left(\bigvee\nolimits_{1\le i\le n} \sim x_i\right). \end{aligned} \tag{22}$$

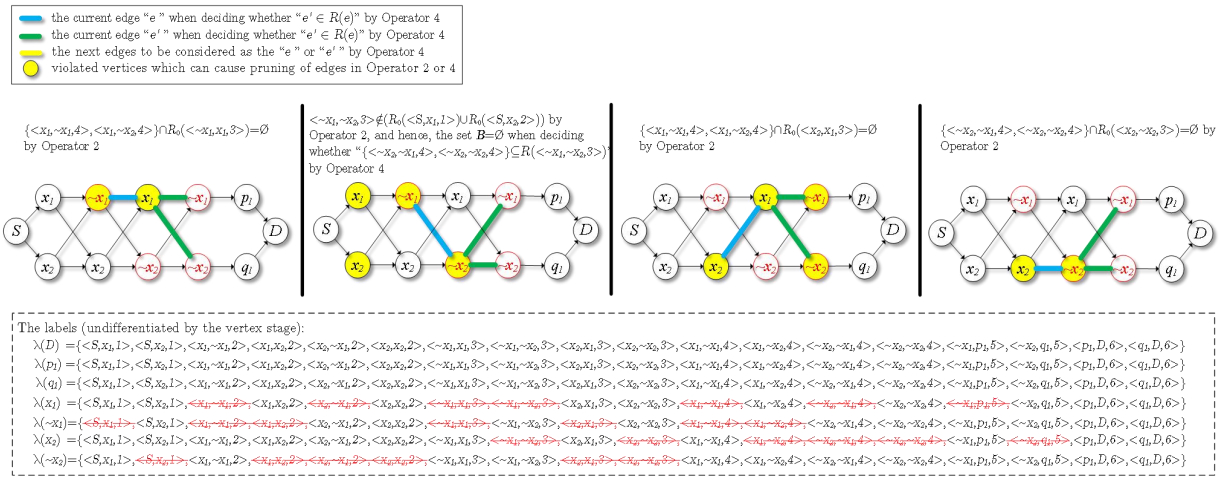


Figure 12: 2-SAT MU running instance

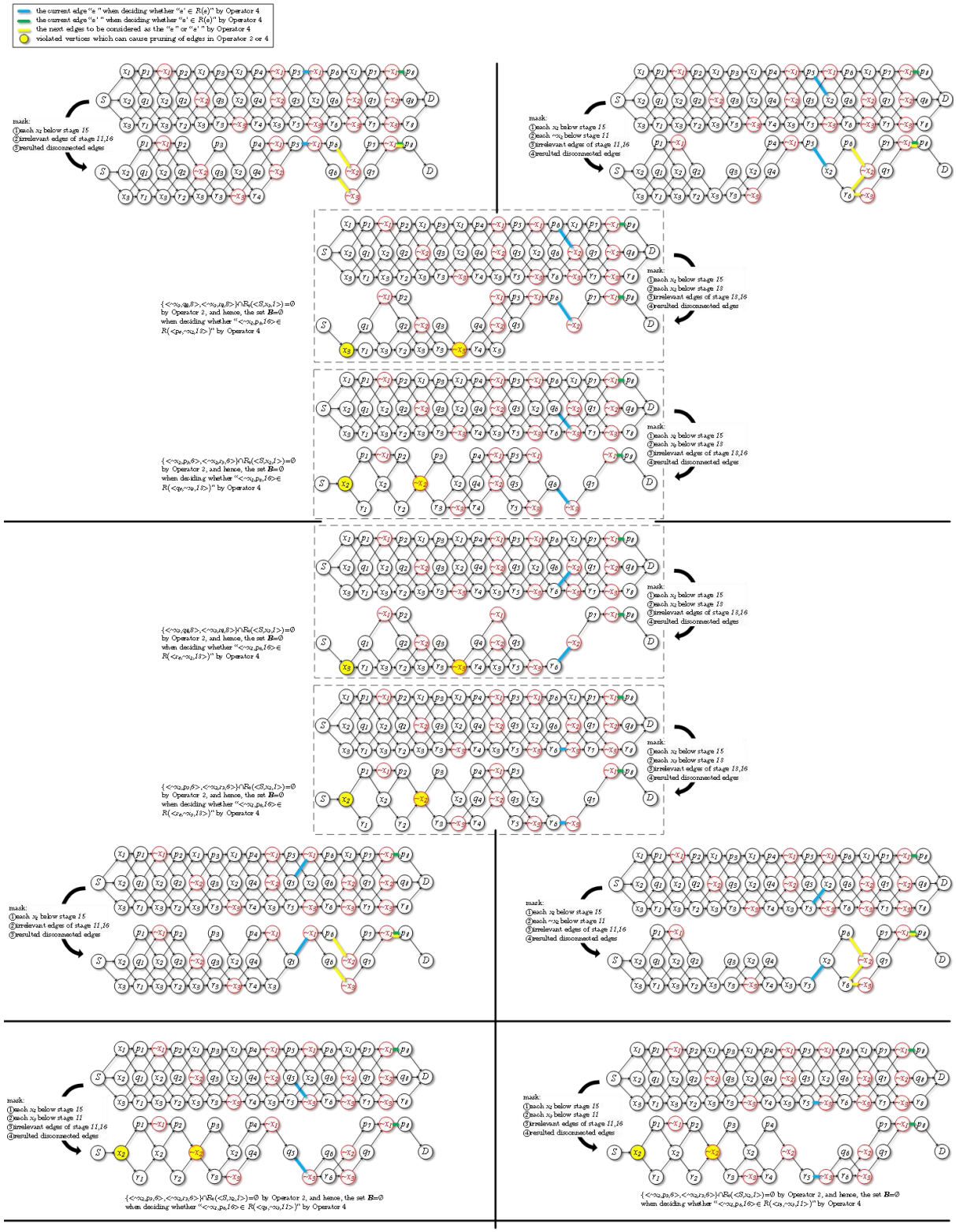


Figure 13: 3-SAT MU running instance

By the definition of $2-\text{MSP}$, we just focus on F_2, F_3.[23] The conversion from $3-\text{SAT}$ to $2-\text{MSP}$ is suggested in Section 5, with auxiliary vertices "p_i","q_i","r_i". Although $2-\text{SAT} \in \text{P}$, we include the instance (Figure 12) might as well to obtain a more comprehensive perspective.

Take F_3 (Figure 13) for example.

All edges of stage 16 can be pruned from each of $R(\langle p_5, \sim x_1, 11\rangle)$, $R(\langle p_5, x_2, 11\rangle)$, $R(\langle q_5, \sim x_1, 11\rangle)$, $R(\langle q_5, \sim x_3, 11\rangle)$, $R(\langle r_5, x_2, 11\rangle)$, $R(\langle r_5, \sim x_3, 11\rangle)$ by Operator 4. This results in $\chi^D_{R(E)}(\lambda(D)) = \emptyset$ and thus a decision of unsatisfiability can be made. Note that the pruning of edges might even happen earlier than as depicted, while the latter makes it easier for illustration without changing the result.

To better focus on the computation of set $\boldsymbol{A}$ and $\boldsymbol{B}$ in Operator 4, irrelevant edges are masked (we did not actually prune these edges from the labeled multi-stage graphs). For example, when deciding whether "$\langle \sim x_1, p_8, 16\rangle \in R(\langle p_5, x_2, 11\rangle)$" by Operator 4: (i) each edge below stage 15 which violates the label of the vertex $\sim x_1$ on $\langle \sim x_1, p_8, 16\rangle$ is masked (see the definition of set $\boldsymbol{A}$ when defining Operator 4); (ii) each edge below stage 11 which violates the label of the vertex x_2 on $\langle p_5, x_2, 11\rangle$ is masked (see the definition of set $\boldsymbol{B}$ when defining Operator 4); and (iii) each edge which deviates paths like $S - \cdots - p_5 - x_2 - \cdots - \sim x_1 - p_8 - D$ is masked (see the definition of set $\boldsymbol{C}$ when defining Operator 4).

Each literal and each clause is indispensable and responsible for the unsatisfiability of the MU formulas; with any missing of single piece of information, a poly-time algorithm might crush into a wrong decision. However, the ZH algorithm, which obviously is not specially designed to solve this very type of instances, is shown to be able to collect all necessary global information for a correct decision, by utilizing the computed stable set $R(E)$ which fulfills the constraint imposed by Operator 4. When deciding whether "$\langle \sim x_1, p_8, 16\rangle \in R(\langle p_5, \sim x_1, 11\rangle)$" by Operator 4, $\langle p_6, \sim x_2, 13\rangle$ can be masked during the computation of the set $\boldsymbol{B}$, since we have $\langle \sim x_1, p_8, 16\rangle \notin R(\langle p_6, \sim x_2, 13\rangle)$ by some earlier step of the ZH algorithm.

15 Concluding remarks

The NP-complete MSP problem, the poly-time ZH algorithm for it, and the correctness proof are introduced in the paper. The result implies $\text{NP} = \text{P}$ and hence finally puts an end to the long-standing well-known problem in computational complexity.

References

[Aar18] Scott Aaronson. HTTPS/Kurtz/eclipse/Charlottesville/Blum/P vs. NP[EB/OL]. *https://scottaaronson.blog/?p=3409*, visited on 12 October 2023.

[ABC+98] David Applegate, Robert Bixby, Vašk Chvátal, and William Cook. 1998. On the solution of traveling salesman problems[J]. *Documenta Mathematica*, Extra Volume ICM III: 645-656.

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. 2004. PRIMES is in P[J]. *Annals of Mathematics*, 160(2): 781-793.

[23] Some extra processing (e.g., Tseitin's Transformation [Tse68]) is needed for $n > 3$, which will make the instance become too large to be illustrated here. For example, when $n = 3$, the corresponding $3-\text{SAT}$ instance generated by Tseitin's Transformation will contain 64 clauses, not even including the extra stages of "virtual" vertices to be added by the reduction to $2-\text{MSP}$.

[Aro98] Sanjeev Arora. 1998. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems[J]. *Journal of the ACM*, 45(5): 753-782.

[AW08] Scott Aaronson and Avi Wigderson. 2008. Algebrization: a new barrier in complexity theory[C]. In *Proceedings of the 40th ACM Symposium on the Theory of Computing*, 731-740.

[Bab16] László Babai. 2016. Graph isomorphism in quasipolynomial time[C]. In *Proceedings of the 48th ACM Symposium on the Theory of Computing*, 684–697.

[BDI21] Markus Bläser, Julian Dörfler, and Christian Ikenmeyer. 2021. On the complexity of evaluating highest weight vectors[C]. In *Proceedings of the 36th Computational Complexity Conference*, 29: 1-36.

[BHK09] Andreas Björklund, Thore Husfeldt, and Mikko Koivisto. 2009. Set partitioning via inclusion-exclusion[J]. *SIAM Journal on Computing*, 39(2): 546-563.

[BI11] Peter Bürgisser, and Christian Ikenmeyer. 2011. Geometric complexity theory and tensor rank[C]. In *Proceedings of the 43rd ACM Symposium on the Theory of Computing*, 509-518.

[BIP16] Peter Bürgisser, Christian Ikenmeyer, and Greta Panova. 2016. No occurrence obstructions in geometric complexity theory[C]. In *Proceedings of the 57th IEEE Annual Symposium on Foundations of Computer Science*, 386-395.

[Bjö14] Andreas Björklund. 2014. Determinant sums for undirected hamiltonicity[C]. In *Proceedings of the 51st IEEE Annual Symposium on Foundations of Computer Science*, 173-182.

[Blu17] Norbert Blum. 2017. A solution of the P versus NP problem[EB/OL]. *https://arxiv.org/abs/1708.03486v1*.

[Can74] Georg Cantor. 1874. Über eine Eigenschaft des Inbegriffes aller reellen algebraischen Zahlen[J]. *Crelle's Journal*, 77: 258-262.

[Chu36] Alonzo Church. 1936. A note on the Entscheidungs problem[J]. *Journal of Symbolic Logic*, 1(1): 40-41.

[Coo03] Stephen A. Cook. 2003. The importance of the P versus NP question[J]. *Journal of the ACM*, 50(1): 27-29.

[DDS21] Pranjal Dutta, Prateek Dwivedi, and Nitin Saxena. 2021. Demystifying the border of depth-3 algebraic circuits[C]. In *Proceedings of the 62nd IEEE Annual Symposium on Foundations of Computer Science*, 92-103.

[DDX+23] Qingxiu Dong, Li Dong, Ke Xu, Guangyan Zhou, Yaru Hao, Zhifang Sui, and Furu Wei. 2023. Large Language Model for Science: A Study on P vs. NP[EB/OL]. *https://arxiv.org/abs/2309.05689*.

[DF12] Rodney G. Downey and Michael R. Fellows. 2012. Parameterized complexity[M]. *Springer Science & Business Media*.

[DIP19] Julian Dörfler, Christian Ikenmeyer, and Greta Panova. 2019. On geometric complexity theory: multiplicity obstructions are stronger than occurrence obstructions[C]. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming*, 51:1-51:14.

[Epp01] David Eppstein. 2001. Small maximal independent sets and faster exact graph coloring[C]. In *Proceedings of the 7th International Workshop on Algorithms and Data Structures*, 462-470.

[FJP14] Shuo Fan, Xinwen Jiang, and Lihong Peng. 2014. Polynomial-time heuristical algorithms for several NP-complete optimization problems[J]. *Journal of Computational Information Systems*, 10(22): 9707-9721.

[FK13] Fedor V. Fomin and Petteri Kaski. 2013. Exact exponential algorithms[J]. *Communications of the ACM*, 56(3): 80-88.

[For09] Lance Fortnow. 2009. The status of the P versus NP problem[J]. *Communications of the ACM*, 52(9): 78-86.

[For21] Lance Fortnow. 2021. Fifty years of P vs. NP and the possibility of the impossible[J]. *Communications of the ACM*, 65(1): 76-85.

[FSS84] Merrick Furst, James B. Saxe, and Michael Sipser. 1984. Parity, circuits and the polynomial-time hierarchy[J]. *Mathematical Systems Theory*, 17: 13-27.

[GIM+20] Ankit Garg, Christian Ikenmeyer, Visu Makam, Rafael Oliveira, Michael Walter, and Avi Wigderson. Search problems in algebraic complexity, GCT, and hardness of generators for invariant rings[C]. In *Proceedings of the 35nd Computational Complexity Conference*, 1-17.

[GJ79] Michael R. Garey and David S. Johnson. 1979. Computers and intractability: a guide to the theory of NP-completeness[M]. *San Francisco: freeman*.

[Göd31] Kurt Gödel. 1931. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I[J]. *Monatshefte für Mathematik Physik*, 38: 173-198.

[Gra04] Andrew Granville. 2004. It is easy to determine whether a given integer is prime[J]. *Bulletin (New Series) of the American Mathematical Society*, 42(1): 3-38.

[GW95] Michel X. Goemans and David P. Williamson. 1995. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming[J]. *Journal of the ACM*, 42(6): 1115-1145.

[Hak85] Armin Haken. 1985. The intractability of resolution[J]. *Theoretical Computer Science*, 39: 297-305.

[HK62] Michael Held and Richard M. Karp. 1985. A dynamic programming approach to sequencing problems[J]. *Journal of the Society for Industrial and Applied Mathematics*, 10(1): 196-210.

[HS78] Ellis Horowitz and Sartaj Sahni. 1978. Fundamentals of computer algorithms[M]. *Computer Science Press*.

[IP16] Christian Ikenmeyer and Greta Panova. 2016. Rectangular Kronecker coefficients and plethysms in geometric complexity theory[C]. In *Proceedings of the 57nd Annual ACM SIGACT Symposium on Theory of Computing*, 396-405.

[IK20] Christian Ikenmeyer and Umangathan Kandasamy. 2020. Implementing geometric complexity theory: on the separation of orbit closures via symmetries[C]. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, 713-726.

[Jia20] Xinwen Jiang. 2020. Polynomial-time algorithm for Hamilton Circuit problem[J], *Computer Science*, 47(7): 8-20 (in Chinese with English abstract).

[JLW+14] Xinwen Jiang, Wanwei Liu, Tianjun Wu, and Litao Zhou. 2014. Reductions from MSP to SAT and from SUBSET SUM to MSP[J]. *Journal of Computational Information Systems*, 10(3): 1287-1295.

[JPW10] Xinwen Jiang, Lihong Peng, and Qi Wang. 2010. MSP problem: its NP-completeness and its algorithm[C]. In *Proceedings of the 5th IEEE International Conference on Ubiquitous Information Technologies and Applications*, 1-5.

[Kar72] Richard M. Karp. 1972. Reducibility among combinatorial problems[J]. *Complexity of Computer Computations*, 85-103.

[Knu02] Don Knuth. 2002. All questions answered[J]. *Notices of the AMS*, 49(3): 318-324.

[Law76] E. L. Lawler. 1976. A note on the complexity of the chromatic number problem[J]. *Information Processing Letters*, 5(3): 66-67.

[Lev86] Leonid A. Levin. 1986. Average case complete problems[J]. *SIAM Journal on Computing*, 15: 285-286.

[LLR+85] E. L. Lawler, Jan Karel Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys. 1985. The traveling salesman problem: a guided tour of combinatorial optimization[M]. *Wiley-Interscience*.

[Mel07] Dieter van Melkebeek. 2007. A survey of lower bounds for satisfiability and related problems[J]. *Foundations and Trends in Theoretical Computer Science*, 197-303.

[MS01] Ketan D. Mulmuley and Milind Sohoni. 2001. Geometric complexity theory I: an approach to the P vs. NP and related problems[J]. *Siam Journal on Computing*, 31(2): 496-526.

[Mul12] Ketan D. Mulmuley. 2012. The GCT program toward the P vs. NP problem[J]. *Communications of the ACM*, 55(6): 98-107.

[MW18] Cody Murray and Ryan Williams. 2018. Circuit lower bounds for nondeterministic quasi-polytime: an easy witness lemma for NP and NQP[C]. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, 890-901.

[Ram14] Srinivasa Ramanujan. 1903-1914. Second notebook (unpublished)[M]. Chapter VI.

[Raz85] Alexander A. Razborov. 1985. Lower bounds on the monotone complexity of some boolean functions[J]. *Soviet Mathematics–Doklady*, 31: 485-493.

[Raz89] Alexander A. Razborov. 1989. On the method of approximations[C]. In *Proceedings of the 21st ACM Symposium on the Theory of Computing*, 167-176.

[RR97] Alexander A. Razborov and Steven Rudich. 1997. Natural proofs[J]. *Journal of Computer and System Sciences*, 55(1): 24-35.

[SAT23] The international SAT competitions web page[EB/OL]. *http://www.satcompetition.org*, visited on 11 February 2023.

[TJR75] Baker Theodore, Gill John, and Solovay Robert. 1975. Relativizations of the P = NP question[J]. *SIAM Journal on Computing*, 4(4): 431-442.

[Tar87] Éva Tardos. 1987. The gap between monotone and non-monotone circuit complexity is exponential[J]. *Combinatorica*, 7(4): 141-142.

[Tse68] Grigori S. Tseitin. 1968. On the complexity of derivation in propositional calculus[J]. *Zapiski nauchnykh seminarov LOMI*, 8: 234-259.

[TSZ10] Olga Tveretina, Carsten Sinz, and Hans Zantema. 2010. Ordered binary decision diagrams, Pigeonhole Formulas and beyond[J]. *Journal on Satisfiability Boolean Modeling and Computation*, 7(1):35-58.

[Tur36] Alan M. Turing. 1936. On computable numbers, with an application to the Etscheidungs problem[C] . In *Proceedings of the London Mathematical Society*, 42: 230-265.

[Val02] Leslie G. Valiant. 2002. Quantum circuits that can be simulated classically in polynomial time[J]. *SIAM Journal on Computing*, 31(4): 1229-1254.

[Vio18] Emanuele Viola. I Believe P = NP[EB/OL]. *https://emanueleviola.wordpress.com/2018/02/16/i-believe-pnp/*, visited on 29 September 2022.

[Wil05] Ryan Williams. 2005. A new algorithm for optimal 2-constraint satisfaction and its implications[J]. *Theoretical Computer Science*, 348(2-3): 357-365.

[Wil14] Ryan Williams. 2014. Nonuniform ACC circuit lower bounds[J]. *Journal of the ACM*, 61(1): 1-32.

[Woe03] Gerhard J. Woeginger. 2003. Exact algorithms for NP-hard problems: a survey[M]. *Combinatorial Optimization—Eureka, You Shrink!*, Lecture Notes in Computer Science, 2570: 185-207.

[Wog22] Gerhard J. Wöginger. The P-versus-NP page[EB/OL]. *https://www.win.tue.nl/~gwoegi/P-versus-NP.htm*, visited on 7 January 2022.

[XBH+07] Ke Xu, Frédéric Boussemart, Fred Hemery, and Christophe Lecoutre. 2007. Random constraint satisfaction: easy generation of hard (satisfiable) instances[J]. *Artificial Intelligence*, 171(8-9): 514-534.

[XL00] Ke Xu and Wei Li. 2000. Exact phase transitions in random constraint satisfaction problems[J]. *Journal of Artificial Intelligence Research*, 12(1): 93-103.

[XG23] Ke Xu and Guangyan Zhou. 2023. SAT Requires Exhaustive Search[EB/OL]. *https://arxiv.org/abs/2302.09512*.