ISSN 1064-5624, Doklady Mathematics, 2020, Vol. 102, No. 2, pp. 376–379. © Pleiades Publishing, Ltd., 2020.
Russian Text © The Author(s), 2020, published in Doklady Rossiiskoi Akademii Nauk. Matematika, Informatika, Protsessy Upravleniya, 2020, Vol. 494, pp. 26–29.

MATHEMATICS

An Almost Exact Linear Algorithm for Transformation of Chain-Cycle Graphs with Optimization of the Sum of Operation Costs

K. Yu. Gorbunov[a,*] and V. A. Lyubetsky[a,b,]**

Presented by Academician of the RAS A.L. Semenov May 14, 2020

Received May 17, 2020; revised August 20, 2020; accepted August 28, 2020

Abstract—For weighted directed chain-cycle graphs, an algorithm transforming one graph into another is constructed. The algorithm runs in linear time and yields a sequence of transformations with the smallest, up to an additive error, total cost. The costs of the operations of inserting and deleting an edge segment may differ from each other and from the costs of the other operations. The additive error is estimated in terms of the operation costs.

Keywords: exact algorithm, graph transformation, 2-degree graph, chain-cycle graph, operation cost, *DCJ*-operations, discrete optimization

DOI: 10.1134/S1064562420050324

In this work, we describe an algorithm that effectively solves the following problem.

Problem. Suppose that a and b are given directed graphs with vertex degrees 1 or 2 and each edge is assigned a name, which is a positive integer. Any vertex of the graph is regarded as obtained by gluing (identifying) the ends of neighboring edges. There are six fixed operations, each assigned a cost, which is a strictly positive number.

The task is to find a sequence of operations of minimum total cost (called a shortest sequence) that transforms a into b. The operations are as follows:

(i) *Del* deletes a connected edge segment with names from a, but not from b, and vice versa.

(ii) *Ins* inserts an edge segment with names from b, but not from a. If a deletion operation results in two free ends (i.e., two vertices of degree 1), they are glued together, while, in the case of insertion, the vertex is first cut (unless it is terminal) and the resulting free ends are then glued at two places.

These operations are a usual deletion of a subword and the insertion of a word as a subword.

(iii) *Cut* cuts a vertex or vice versa.

(iv) *OM* glues two free ends.

Operation (iii) applied to a single vertex of degree 2 yields two vertices of degree 1, while operation (iv) applied to two vertices of degree 1 produces one vertex of degree 2. The following two operations are compositions of the preceding ones.

(v) *SM* cuts a vertex and glues one of the resulting free ends to a previously available free end.

(vi) *DM* cuts two vertices and glues the resulting free ends.

The last four operations are collectively called *DCJ* operations; they are defined in [1].

This completes the formulation of the problem.

Previous results concerning this problem, including its applied aspects, are overviewed in [2, Chapter 10]. In all previous works by other authors, of which we note the latest one [4], the problem was solved assuming identical costs of *DCJ*-operations and identical costs of *Del* and *Ins* operations, which substantially simplifies the problem as compared with the assumptions made in Theorem 1 below. Theorem 1 states that the algorithm described after it runs in linear time and is exact up to an additive constant. Let the costs of *Del* and *Ins* operations be denoted by w_d and w_i, respectively.

Theorem 1. *If the DCJ-operations are assigned equal costs w, then the algorithm generates a sequence with a*

[a] *Institute for Information Transmission Problems of the Russian Academy of Sciences (Kharkevich Institute), Moscow, 127051 Russia*

[b] *Faculty of Mechanics and Mathematics, Lomonosov Moscow State University, Moscow, 119991 Russia*

**e-mail: gorbunov@iitp.ru*

***e-mail: lyubetsk@iitp.ru*

total cost different from its minimum value by at most an additive constant k depending only on the operation costs.

Namely, if $\max\{w_d, w_i\} \le w$, then $k = 0$; if $\min\{w_d, w_i\} \ge w$, then $k = 2w$; and if $\min\{w_d, w_i\} < w < \max\{w_d, w_i\}$, then $k = w_i - 1$ (if $w_d + w_i \le 2w$), $k = 4w_i + 2w_d - 6$ (if $w_d + w_i > 2w$ and $\max\{w_d, w_i\} \le 2w$), and $k = 6w_i + 2w_d - 9$ (if $\max\{w_d, w_i\} > 2w$).

The proof of Theorem 1 implies that, in fact, k (if nonzero) can be reduced noticeably, but the corresponding more complicated expressions are omitted.

In the rest of this paper, we describe the algorithm and its properties. Let us begin with auxiliary definitions.

Recall the definition of the graph $a + b$ from [5]. Other versions can be found in earlier works, for example, in [6].

Definition [5]. The *vertices* in $a + b$ are all ends of edges that belong to both a and b and, additionally, vertices that are uniquely assigned to each maximum connected edge segment ("block") in $a\backslash b$ or $b\backslash a$; the vertices are labeled a or b, respectively.

The *edges* in $a + b$ join vertices if the latter are glued in a or b or the end of a block is glued to a vertex in a or b; the edges are labeled a or b, respectively.

In $a + b$, a *singular* vertex is a vertex inside the pair of edges aa or bb, a labeled end of a chain, or a labeled isolated vertex; the other vertices are call *ordinary*. An edge with a singular end is called *singular*; the other edges are referred to as *ordinary*.

A graph is said to be of final form if it consists of ordinary isolated vertices and cycles of length 2, each having one edge labeled a and the other labeled b. In $a + b$, the cost of a deletion becomes the cost w_a of deleting an a-singular vertex, while the cost of an insertion becomes the cost w_b of deleting a b-singular vertex. The *DCJ*-operations remain the same, except the deletion of a segment is replaced by the deletion of an a-singular vertex and the insertion of a segment is replaced by the deletion of a b-singular vertex.

An edge is called pendant if it is incident to a singular vertex of degree 1. The size of a component in the graph $a + b$ is the number of its ordinary edges plus half the number of its singular nonpendant edges. The size is equal to 0 for ordinary isolated vertices and loops and is equal to -1 for singular isolated vertices. A chain of odd (even) size is called odd (even). Suppose that a chain does not contain ordinary edges. The types of such chains are defined as follows. Type $1a$ is an odd chain with a single pendant b-edge. Type $2a^*$ is an odd chain with two pendant b-edges. Type $2a'$ is a b-singular isolated vertex. Type $2a$ is type $2a^*$ or $2a'$. Type $3a^*$ is an odd chain without pendant edges with two terminal a-edges and a b-singular vertex. Type $3a'$ is an aa chain. Type $3a$ is type $3a^*$ or $3a'$. Type 1_a^* is an even chain with a single pendant a-edge and a b-singular vertex. Type $1_a'$ is a pendant a-edge. Type 1_a is type 1_a^* or $1_a'$. The notation with a replaced by b is similar. Type 2* is an even chain with two pendant nonincident edges. Type 2' is two pendant edges incident to a common ordinary vertex. Type 2 is type 2* or 2'. Type 3 is an even chain without pendant edges, but with singular vertices. Type 0 is a chain without singular vertices. The type of a chain with ordinary edges is defined as the type of the chain obtained after deleting these edges; this definition does not depend on the order of these deletions [5, Lemma 6]. An (a, b)-cycle is a cycle with singular a- and b-vertices. An a-cycle is a cycle with singular a-vertices, but without singular b-vertices. A b-cycle is defined in a similar manner.

The algorithm starts with the graph $a + b$ and sequentially generates graphs G until a graph of final form is obtained. All G in this sequences have the form of $c + d$ for their graphs c and d. A sequence starting with $a + b$ and ending with a graph of final form is called reducing. After the shortest reducing sequence was found for $a + b$, the desired shortest transformation of a into b is constructed from it in linear time.

The algorithm consists of eight stages.

Stage 0. Transform the initial pair of graphs a and b into a new (breakpoint) graph $a + b$. The original problem is equivalent to the problem of reducing this graph $a + b$ to a final form by applying analogues of the original operations with a and b. The proof of the equivalence of these two problems repeats word for word the proof of Corollary 5 in [5], which assumes only the equality of the costs of *DCJ*-operations.

Stage 1. Delete ordinary edges from all components of nonfinal form, i.e., apply DM to a pair of neighboring edges, apply SM if one of the neighboring edges is absent, or apply OM if both neighboring edges are absent.

Stage 2. Perform the found compositions of the original operations, which are called interactions. These compositions are applied to chains, whose type is indicated on the left-hand side of one of the following equalities; the type of the result is indicated on the right-hand side of the equality. Thus, 2-interactions are SM with the following equalities of terms (the chain to be cut is everywhere indicated the first; ordinary isolated vertices are not presented on the right-hand side):

$$1a + 1b = 1_b^*,\ 3a + 2b = 1_a,\ 3b + 2a = 1_b,\ 3 + 2 = 1_b^*,$$
$$(1a + 2b) + 3 = 1_b^*,\ (1b + 2a) + 3 = 1_b^*,\ (3a + 1b) + 2 = 1_b^*,\ (3b + 1a) + 2 = 1_b^*,\ 1a + 2 = 2a^*,\ 1b + 2 = 2b^*,\ 3 + 1a = 3a^*,\ 3 + 1b = 3b^*,\ (3b + 1a) + (1a + 2b) = 1_b^*,\ (3a + 1b) + (1b + 2a) = 1_b^*,\ 1a + (1a + 2b) = 2a^*,\ 1b + (1b + 2a) = 2b^*,\ (3b + 1a) + 1a = 3a^*,\ (3a + 1b) + 1b = 3b^*,\ 1a + 2b = 2,\ 1b + 2a = 2,\ 3a + 1b = 3,\ 3b + 1a = 3,\ 3 + ((3 + 2b) + 2a) = 1_b^*,\ (3a + (3b + 2)) + 2 = 1_b^*,\ (3a + 2) + 2 = 2a^*,\ (3b + 2) + 2 = 2b^*,\ 3 + (3 + 2a) =$$

$3a^*$, $3 + (3 + 2b) = 3b^*$, $(3 + 2b) + 2a = 2^*$, $3a + (3b + 2) = 3$ and OM with the equality $1a + 1a = 3a^*$ or $1b + 1b = 3b^*$. If $\min\{w_d, w_i\} < w$, then perform two more 2-interactions defined as SM: $3a^* + 3b = 3$ or $2a + 2b^* = 2^* + 1'_a$.

Each 2-interaction is applied as long as $\min\{w_d, w_i\} \geq w$; otherwise, a maximum set of interactions with no common arguments is constructed.

These interactions and the ones indicated below strictly reduce the complexity of the current graph G, which is measured by the minimum number $t(G)$ of operations required for the reduction of G. Importantly, $t(G)$ is expressed in terms of simple characteristics of G (except for one, denoted by $P(G)$, which we were nevertheless able to compute, see below), so overall $t(G)$ is easily computed. This is the basic idea underlying both the algorithm and the proof of its exactness.

At Stages 3 and 5–7, each interaction is applied as long as possible.

Stage 3. If $\max\{w_d, w_i\} > w$, then perform the following 3-interactions, which decrease the number of components in G and reduce its complexity: DM (but SM for an isolated vertex): b-loop + "component with a b-singular vertex" = the same component; SM: $2 + 2 = 2 + 1'_a$ (SM cuts off a pendant a-edge and glues the resulting end to an terminal b-singular vertex of another chain); OM: $2a + 2a = 2a$, $2b^* + 2b^* = 2b^*$ (gluing of pendant vertices of two chains); OM and DM:

$3a^* + 3a^* = 3a^*$, $3b + 3b = 3b$ (OM is followed by the deletion of an ordinary edge); SM: $3a^* + 2 = 1a$, $3b + 2 = 1b$; $3 + 2a = 1a$, $3 + 2b^* = 1b$; OM and DM: $1a + 3a^* = 1a$, $1b + 3b = 1b$ (as above); OM: $1a + 2a = 1a$, $1b + 2b^* = 1b$; OM and DM: $3a^* + 3 = 3$, $3b + 3 = 3$ (as above); OM: $2a + 2 = 2$, $2b^* + 2 = 2^*$; SM: $3 + 3 = 3$; $1^*_a + 1^*_a = 1^*_a$, $1_b + 1_b = 1_b$; $1_b + 1a = 1a$, $1^*_a + 1b = 1b$; $1a + 1^*_a = 1a$, $1b + 1_b = 1b$; $1_b + 2a = 2a$, $1^*_a + 2b^* = 2b^*$; $3a^* + 1^*_a = 3a^*$, $3b + 1_b = 3b$; $3 + 1^*_a = 3$, $3 + 1_b = 3$; $1^*_a + 2 = 2^*$, $1_b + 2 = 2$.

3-Interactions are depicted in [3, pp. 8–9].

Stage 4. Close chains of strictly positive size into cycles by applying OM operations (if a chain is odd) or SM operations (if a chain is even; there remains a terminal isolated ordinary vertex or a terminal isolated pendant edge).

Stage 5. If $w_i + w_d > 2w$, then perform the following 5-interactions. Twice DM: (a, b)-cycle + (a, b)-cycle = (a, b)-cycle = (a, b)-cycle + cycle of length 2 (the second DM deletes an ordinary edge); twice DM: (a, b)-cycle of size strictly larger than 2 = (a, b)-cycle of the same size with an ordinary edge = (a, b)-cycle of smaller size + cycle of length 2 (deletion of an ordinary edge). They are shown in figures in [3, p. 9]. If $w_i + w_d \leq 2w$, then apply DM to each cycle of size strictly larger than 2 to cut off a cycle of size 2 with a singular a-vertex (if $w_i \geq w_d$) or with a b-vertex (if $w_i < w_d$).

Stage 6. If $w_i + w_d > 2w$, then perform 6-interactions between an (a, b)-cycle of size 2 and chains of size 0. Twice SM: (a, b)-cycle + $2a' = 1b$ and then $1b + 2b' = 2'$. Twice DM: (a, b)-cycle + $2' = 2'$ with an ordinary edge = $2'$ + cycle of length 2 (deletion of an ordinary edge). They are shown in figures in [3, pp. 9–10].

Stage 7. If $\max\{w_d, w_i\} > 2w$, then perform the following 7-interactions with cycles of size 2 and chains of size 0. Twice DM: (a, b)-cycle + a-cycle = (a, b)-cycle of size 4 with two ordinary edges = (a, b)-cycle + cycle of length 2 (deletion of an ordinary edge). Twice DM: a-cycle + a-cycle = a-cycle of size 4 with three ordinary edges = a-cycle + cycle of length 2 (deletion of an ordinary edge). Twice SM: a-cycle + $2' = 2'$ with a single terminal ordinary edge = $2'$ + cycle of length 2 (deletion of an ordinary edge). SM and OM: (a,b)-cycle + $2b' = 1b = (a, b)$-cycle. Twice SM: (a, b)-cycle + $1'_a = 3 = (a, b)$-cycle. Twice SM: a-cycle + $2b' = 2b'$ with a single terminal ordinary edge = $2b'$ + cycle of length 2 (deletion of an ordinary edge). Twice SM: a-cycle + $1'_a = 1'_a$ with a single terminal ordinary edge = $1'_a$ + cycle of length 2 (deletion of an ordinary edge). Depictions of 7-interactions can be found in [3, p. 10]. With the use of 7-interactions, the deletion of a singular vertex is replaced by two DCJ-operations, which is beneficial if $\max\{w_d, w_i\} > 2w$. These interactions are applied to the case $w_d > 2w$. If $w_i > 2w$, they are applied with a and b interchanged; if both inequalities hold, then interactions of both types are applied.

Stage 8. Delete the singular vertices.

The linear running time of the algorithm follows from the fact that the graph $a + b$ is constructed by searching once through the components in a and b. Additionally, Stage 1 requires that the components in $a + b$ be searched through once. The number of operations executed in the algorithm is linear, since each interaction consisting of at most three operations reduces the number of vertices in $a + b$ and does not increase the number of its edges or reduces the number of edges in the nonfinal part of $a + b$ and does not increase the number of its vertices. Each operation is executed in a constant time.

Let $T(G)$ be the total cost of the operations in the sequence generated by the algorithm on G. The exactness in Theorem 1 is proved using inequalities of the following type: for any initial operation o and any graph G, it is true that $c(o) \geq T(G) - T(o(G))$, where $c(o)$ is the cost of the operation o and $o(G)$ is the result of applying o to G.

FUNDING

This work was supported by the Russian Foundation for Basic Research, project no. 18-29-13037.

REFERENCES

1. S. Yancopoulos, O. Attie, and R. Friedberg, Bioinformatics **21**, 3340–3346 (2005). https://doi.org/10.1093/bioinformatics/bti535

2. *Bioinformatics and Phylogenetics: Seminal Contributions of Bernard Moret*, Ed. by T. Warnow (Springer Nature, Switzerland AG, 2019).

3. K. Yu. Gorbunov and V. A. Lyubetsky, arXiv:2004.14351 [math.CO] (2020).

4. P. H. da Silva, R. Machado, S. Dantas, and M. D. V. Braga, J. IEEE/ACM Trans. Comput. Biol. Bioinf. **14** (3), 1–6 (2017).

5. K. Yu. Gorbunov and V. A. Lyubetsky, Probl. Inf. Transm. **53** (1), 55–72 (2017). https://doi.org/10.1134/S0032946017010057

6. M. A. Alekseyev and P. A. Pevzner, Theor. Comput. Sci. **395** (2–3), 193–202 (2008). https://doi.org/10.1016/j.tcs.2008.01.013

Translated by I. Ruzanova