Graph Transformation Approach for the Shortest Path Search and Length Calculation

Mori, Masao
Department of Informatics, Kyushu University

Mizoguchi, Yoshihiro
Graduate School of Mathematics, Kyushu University

https://doi.org/10.5109/3625

Bulletin of Informatics and Cybernetics, Vol. 00, No. 00, 00

GRAPH TRANSFORMATION APPROACH FOR THE SHORTEST PATH SEARCH AND LENGTH CALCULATION

By

Masao Mori[*] and **Yoshihiro Mizoguchi**[†]

Abstract

We consider a graph with labels of edges. A label means the length of an edge. We present a method to compute the length of the shortest path between two vertices using graph transformations. We introduce graph transformation rules which preserve the length of paths. Reducing to a simple graph which contains two vertices, we finally calculate the length of the shortest path of those two vertices. There were several algorithms for computing network reliabilities using graph transformations. We use the same framework as those algorithms for applying the graph transformation rules, but our transformation rules do not calculate the network reliabilities but calculate the length of the shortest path.

1. Introduction

Graph transformation (graph reduction or graph rewriting) techniques are useful for developing an algorithm to compute several values of graph properties in the area such as circuits and networks.

Politof and Satyanarayana (1986a) introduced a polynomial-time algorithm for the computation of network reliability for some restricted class of graphs, and Satyanarayana and Wood (1985) studied $O(|E|)$-time algorithms for k-terminal reliability for series and parallel graphs. Politof and Satyanarayana (1986b) studied all-terminal reliability for IFCF (inner for cycle free) graphs and they also presented $O(|V|)$-time algorithms for PCF (planar cube free) graphs. In these studies graph transformation methods were introduced to compute network reliabilities. Network graphs were classified into some class, e.g. IFCF graphs and PCF graphs, to give efficient algorithms using graph transformation methods.

Okada and Hayashi (1991) showed the theoretical results about critical pair lemma for the graph transformation rules introduced by Politof and Satyanarayana (1986a; 1986b). The notion of critical pair was firstly considered in the area such as studies of term rewriting systems. In Mizoguchi and Kawahara (1995) and Mizoguchi (1999) the concept of critical pair lemma was generalized to a graph rewriting system, and a complete graph rewriting system was introduced to computes network reliabilities such as Okada and Hayashi (1991).

In this paper, we introduce graph transformation rules to search the shortest path and compute its length for given graphs and construct an $O(|E|)$-time algorithms for

[*] Department of Informatics, Kyushu University, 33 Fukuoka 812-8581, Japan. masa@i.kyushu-u.ac.jp
[†] Graduate School of Mathematics, Kyushu University, 33 Fukuoka 812-8581, Japan. ym@math.kyushu-u.ac.jp

series and parallel graphs. Application of those rules yields to the shortest path and its length for given two vertices. Using those rules, it is easily verified that the method of the 2-terminal network reliability algorithms introduced by Politof and Satyanarayana (1986b) is available to search the shortest path and its length. Especially, by modifying the result of Politof and Satyanarayana (1986a), we present an $O(|E|)$-time algorithm to calculate the length of the shortest path for series and pararell graphs.

We review an invariant property for IFCF and ICF graphs in terms of $\Delta - Y$ graph transformation rules. Further, we reconstruct $\Delta - Y$ graph transformation rules for computing the length of the shortest path. This construction shows a possibility to make an efficient algorithm using graph transformations for computing the length of the shortest path. Politof and Satyanarayana's algorithms for IFCF graphs and PCF graphs compute an all-terminal reliability of networks but not 2-terminal reliability, so it is not obvious to apply their results to calculate the length of the shortest path between two vertices.

2. Preliminary

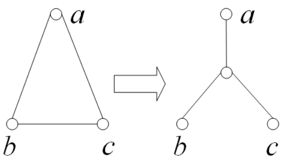
We consider a graph $G = (V, E)$ (where V and E are the sets of vertices and edges of G). Each edge $e \in E$ has a label $l(e) \geq 0$ which indicates the length of the edge.

A *cutvertex* of a graph is a vertex whose removal disconnects the graph. A *non-separable*(or *biconnected*) graph is a connected graph with a cutvertex. A *separation pair* of a graph is a pair of vertices whose removal disconnects the graph. If a graph contains no separation pairs, it is called *triconnected*. Distinguished edges with the same vertices are called the *parallel* edges. If two edge is connected at a vertex whose degree is 2, then the edge is called the *series* edges. A *plane* graph can be drawn on the plane. A plane graph partitions the plane into a number of connected regions; the closures of these regions are called the *faces*. With respect to any face f the vertices of the graph can be distingusied into two sets: the *inner* (written $I_f(G)$) and *outer* (written $O_f(G)$) vertices of G.

A plane graph G with no series or parallel edges is said to be an *inner-cycle-free*(*ICF-graph*), if there exists a face f in G such that every cycle of G contains at least one vertex of f. A plane graph G with no series or parallel edges is said to be an *inner-four-cycle-free*(*IFCF-graph*), if there exists a face f in G such that every cycle with 4 or more edges of G contains at lease one vertex of f. Those faces are referred to as *window* of G. The subgraph induced by all inner vertices of graph G with respect to a window w will be written in $I_w(G)$ or I_w in short if the graph G is trivial from the context. An ICF-graph is called a *wheel graph* if it has a window with respect to which the graph has exactly one major inner vertex.

3. ICF-graph and $\Delta - Y$ replacements

In this section we review an invariant property of IFCF-graphs and ICF-graphs in terms of $\Delta - Y$ replacements Politof and Satyanarayana (1986b). We start to review a procedure to recognize IFCF-graphs with no series or parallel edges. Suppose that G is an IFCF-graph and w is a window. If G has no series and no parallel edges, then every cycle of G that does not contain any vertices of w must have exactly three edges, that is, Δ. Application of $\Delta - Y$ replacements on such Δ yields to an IFCF-graph with respect to w again. The class of ICF-graphs includes IFCF-graphs. The same property as above is obtained. That is, we have an invariant property of ICF-graphs in terms of $\Delta - Y$ replacements. Let $G = (V, E)$. We will test whether G is an IFCF-graph. This procedure requires $O(|V|^2)$ operations. $\Delta - Y$ graph transformation can be shown in the following figure.



Procedure 1 Recognition algorithm of IFCF-graphs
Input : *A triconnected graph $G = (V, E)$ with no series or parallel edges.*
Output : *Answer whether G is IFCF-graph or not. If yes, the window will be also output.*

step 1 *Chech whether G is planner. If G is planner, obtain a planner embedding of G. Otherwise STOP.*

step 2 *Label the faces of G, $f_1, f_2, \ldots, f_{|E|-|V|+2}$ arbitrarily. Initialize $i \leftarrow 1$.*

step 3 *Partition vertices into inner and outer vertices with respect to f_i.*

step 4 *Consider the subgraph I_{f_i} Find the maximal biconnected components(block) of I_{f_i}. If every block contains 3 or fewer vertices then G is IFCF-graph with respect to f_i; hence RETURN yes with $w \leftarrow f_i$. Otherwise increment $i \leftarrow i + 1$.*

step 5 *If $i > |E| - |V| + 2$ then STOP; RETURN no. Otherwise go to step 3.*

Now we can obtain IFCF-graphs from triconnected graphs. A 3 cycle with no vertices of a window in IFCF-graph is called *inner* Δ. We derive the following property from the definition of IFCF-graphs.

Proposition 3.1. *If IFCF-graph G with respect to a window w does not have inner-Δs, then G is a ICF-graph with respect to w.*

Proposition 3.2. *Suppose that IFCF-graph G with respect to a window w has inner-Δs. If f is an inner-Δ in G and G' is the graph obtained from G by replacing f with a Y, then G' also is an IFCF-graph with respect to w.*

Proof. Suppose that G' is not IFCF-graph. Since G is IFCF-graph, there must be a cycle C not containing vertices of w in G' which have more than 4 vertices. Clearly C contains a new added vertex v by $\Delta - Y$ replacement because G is IFCF-graph. Then we have a cycle in G consisting of more than 4 vertices of Δ and C except v. This cycle does not contain any vertex of the window. Contradiction.

These previous propositions implies that ICF-graph can be obtained from IFCF-graph by $\Delta - Y$ replacement. Now we present the following procedure.

Procedure 2 Algorithm to obtain ICF-graphs from IFCF-graphs
Input : *IFCF-graph $G = (V, E)$ having no series or parallel edges, and window w of G.*
Output : *ICF-graph G' with respect to w.*

step 1 *Partition vertices into inner vertices I_w and outer vertices O_w.*

step 2 *Consider the subgraph I_w induced by the inner vertices. Find the blocks of I_w. Every block with exactly three vertices constitutes inner Δ.*

step 4 *Replace all inner Δs by Ys. The resulting graph is an ICF-graph with respect to w.*

Proposition 3.3. *Suppose that non-series and non-parallel graph G is a ICF-graph with respect to w. If G is triconnected, then the subgraph I_w induced inner vertices of G is a tree.*

Proof. By the definition of ICF-graph, I_w does not have any cycle. It is sufficient to show that I_w is connected. Suppose otherwise. Choose two disconnected vertices u and v from I_w. In the graph G every path from u to v contains outer vertices. There must be at most two disjoint pathes since G is planner. This contradicts the fact that G is triconnected. Hence I_w is connected.

Proposition 3.4. *Let G be a triconnected plane ICF-graph with no series or parallel edges, and let w be a window of G. Suppose that u is a leaf of I_w. Then G contains a Δ formed by u and outer vertices of G.*

Proof. By the proposition 3.3 I_w is a tree. If $|I_w| = 1$, G is a wheel graph since G is non series graph. Then there must exist a Δ whose vertices are u and two outer vertices. Since u is a leaf of I_w, u connects to one inner vertex and connects at least two outer vertices. If those outer vertices are adjacent, then u and those outer vertices forms Δ. Suppose otherwise. Let v be a vertex between those outer vertices. Since the graph is planner and non series, u and v are not adjacent. Then those outer two vertices could be a separation pair. This contradicts.

Theorem 3.5. *If G is planner ICF graph and non series and non paralell and w is window of G, then*

1. *G contains $\Delta \neq w$,*
2. *any $\Delta - Y$ replacement in G yields a ICF graph where $\Delta \neq w$.*

Proof. We firstly prove 1. Obviously G has at least one inner vertex with respect to w. If G is triconnected, then G contains $\Delta \neq w$ by proposition 3.4. Assume that G is biconnected but not triconnected. Let (v_1, v_2) be a separation pair of G. We obtain a triconnected component G' of G by adding the edge (v_1, v_2). As G' is easily shown to be ICF graph with respect to some window, the subgraph g' induced by the inner vertices of G' is tree by proposition 3.3. By proposition 3.4, a leaf of g' and outer vertices of G' forms a Δ which does not contain the edge (v_1, v_2).

Prove 2. Let the new vertex v by applying $\Delta - Y$ transformation and let G' be the transformed graph. We need two cases. Firstly consider $|\Delta \cap w| = 1$. If a cycle containing v and two inner vertices in Δ does not include vertices in w, then G has a cycle containing two inner vertices in Δ and it also does not include vertices in w. This contradicts the assumption. Secondly assume $|\Delta \cap w| = 2$. Obviously every cycle containing v includes outer vertices in Δ.

In Politof and Satyanarayana (1986b) the theorem 3.5 ensures that series and parallel graphs can be obtained by applying $\Delta - Y$ transformation to ICF graphs appropreately.

4. Rules which preserve length of the shortest path

A graph after making all series and parallel replacements showed in Figure 1 from a graph with a single edge is called **series-parallel graph**.

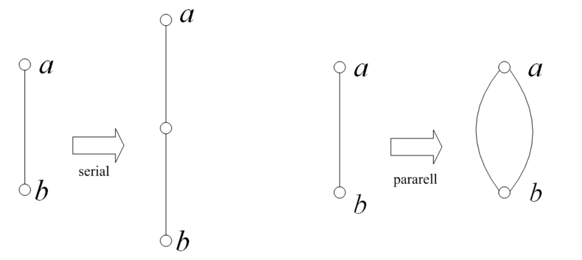


Figure 1: series replacement, pararell replacement

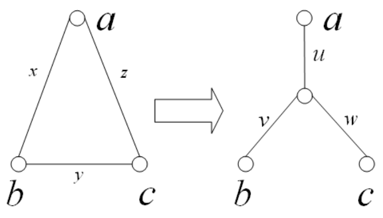


Figure 2: $\Delta - Y$ transformation

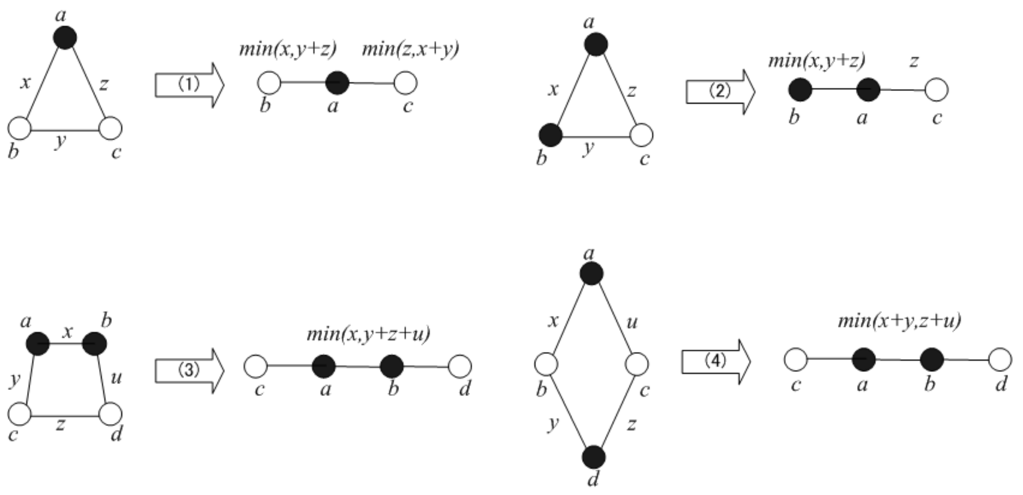


Figure 3: polygon to chain transformations

Consider a graph G and a set of two vertices $K = \{s, t\}$. Two vertices indicate a starting vertex and a goal vertex to compute the length of the shortest path. Following two graph transformation rules(serial transformations and parallel transformations) are called **simple transformations**.

Series transformation rule: For a pair of edges $e_1(a, b)$, $e_2(b, c)$ ($a \neq c$, $deg(b) = 2$, $b \notin K$), remove a vertex b and two edges e_1 and e_2 and insert a single edge $e_3 = (u, w)$ where $l(e_3) = l(e_1) + l(e_2)$.

Parallel transformation rule: Replaces a pair of edges $e_1 = (a, b)$ and $e_2(a, b)$ with a single edge $e_3(a, b)$ where $l(e_3) = \min(l(e_1), l(e_2))$

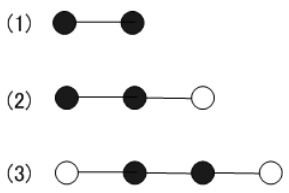


Figure 4: terminal pattern graphs

We construct more several transformation rules to compute the length of the shortest path.

Δ-Y transformation rule is a rule showed in Figure 2 where $p = min(x, y + z)$, $q = min(y, z+x)$, $r = min(z, x+y)$ and $u = \frac{p-q+r}{2}$, $v = \frac{p+q-r}{2}$, $w = \frac{-p+q+r}{2}$. Four rules showed in Figure 3 are called **polygon to chain rules**. We will apply polygon to chain rules to a graph when we cannot apply any simple transformation rules.

We call one of three pattern of graphs showed in Figure 4 as a **terminal pattern graph**. A terminal pattern graph is not able to be applied any transformation rules simple rules, Δ-Y rules and polygon to chain rules. We note that a black vertex in Figure 3 and Figure 4 is a vertex in a set $K = \{s, t\}$.

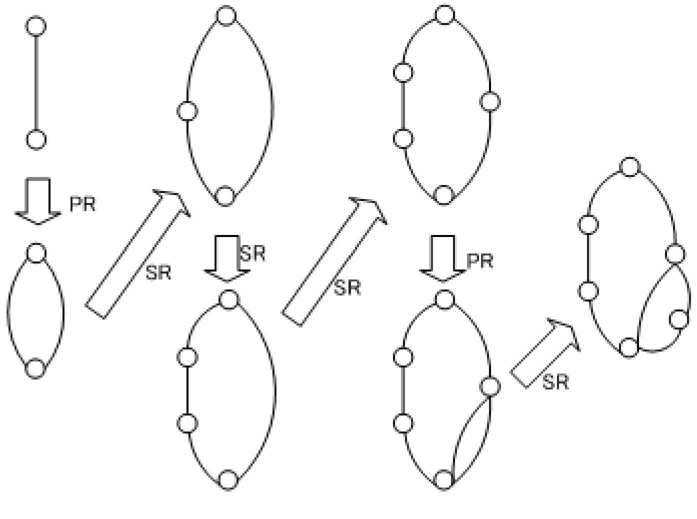


Figure 5: Series parallel graph generation

The next proposition shows that our defined graph transformation rules preserve the length of the shortest path. So we can make an algorithm for computing the length of the shortest path using our graph transformation rules

Proposition 4.1. *Let G be a series parallel graph G, K a set of two vertices in G. If a graph G' is an obtained graph by applying one of simple rules, $\Delta - Y$ rule or*

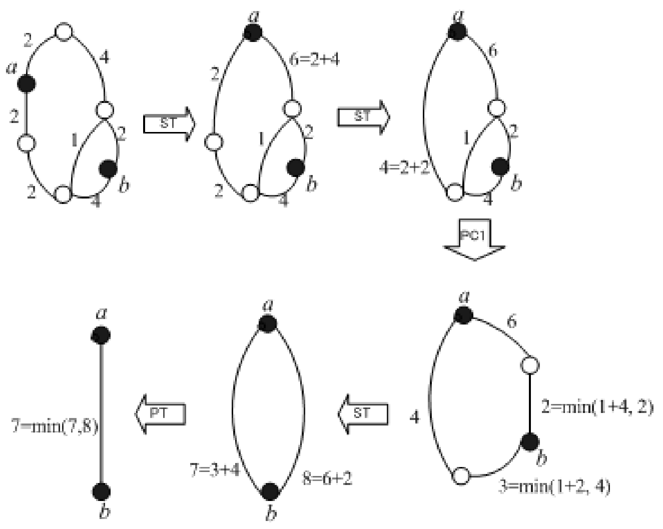


Figure 6: Calculation using graph transformations

polygon to chain rules to G, then the length of the shortest path between two vertices in K of G is equal to its of G'.

The next two proposition guarantee that we can compute the length of the shortest path for a series parallel graph by just applying our tranformation rules.

PROPOSITION 4.2. *Let G be a series parallel graph G, K a set of two vertices in G, G' an obtained graph by applying one of simple rules or polygon to chain rules to G. A graph G is a series parallel graph if and only if a graph G' is a series parallel graph.*

PROPOSITION 4.3. *Let G be a series parallel graph G, K a set of two vertices in G. If G is not a terminal pattern graph, there exists a simple rule or a polygon to chain rule which is applicable to G.*

Using a similar algorithm introduced by Satyanarayana and Wood (1985), we obtain the next theorem.

THEOREM 4.4. *Let G be a noseparable series-parallel graph. Then for any two vertices, the length of the shortest path is computable in $O(|E|)$ time.*

Now we claim the algorithm for the shortest path calculation.

Procedure 3 The Shortest Path

```
Input  :  A nonseparable series-parallel graph $G = (V, E)$
       :  where $|V| \geq 2$ and $|E| \geq 2$, $K = \{s, t\}$.
Output :  The length of the shortest path from $s$ to $t$ in $G$
```

STEP 1 *For each $v \in V$ $(deg(v) = 2)$ perform series transformations.*

STEP 2 *For each $v \in V$ $(deg(v) > 2)$ perform polygon-to-chain transformations.*

Up to 2 times of series transformations.

The main subject of this paper is to intorduce the rewriting rules for calculating the shortest path length and that the proof for the properties about applying rewriting rules is induced by similar techniques in Satyanarayana and Wood (1985). We omit the proof but we show two examples, a graph generation using series and parallel replacements (Figure 5) and a calculation of the shortest path length using graph transformations (Figure 6).

Searching the shortest path can be done concurrently by adding path information to label of edge (costs) and rewriting rules for the path informations. Path information is written with blanket, e.g. $[abc]$ and added to the cost of the edge. Initially every edge has no path information. The followings (Figure 7 and Figure 8)are transformation rules with path informations where $l, m, n, \cdots$ are sequences of vertices and ';' shows concatenation.

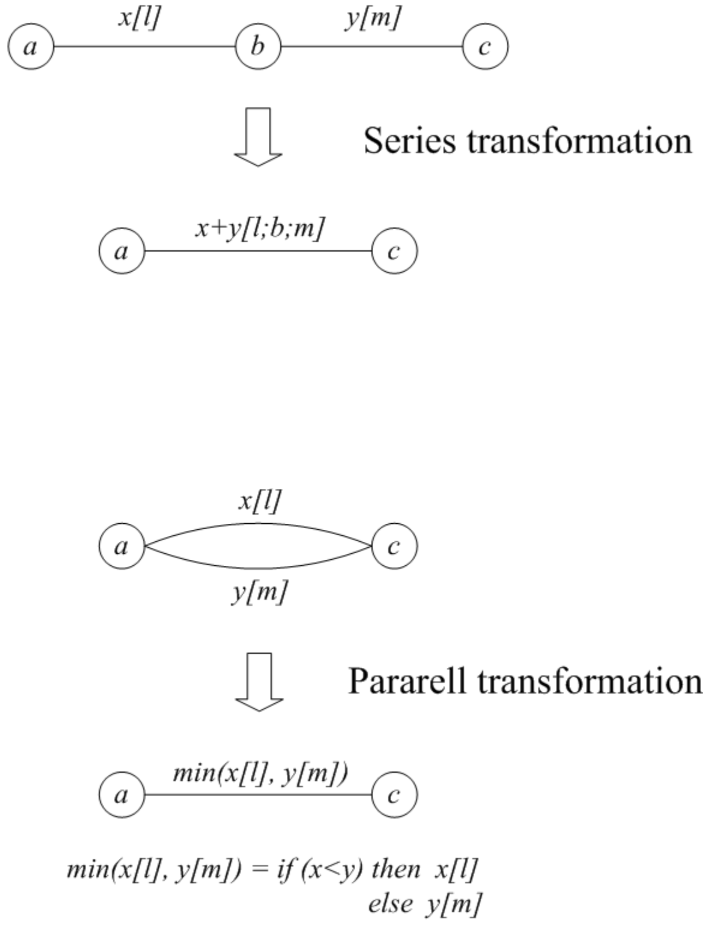
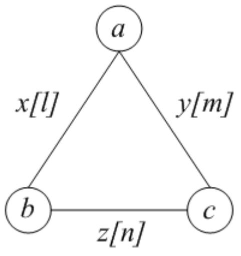
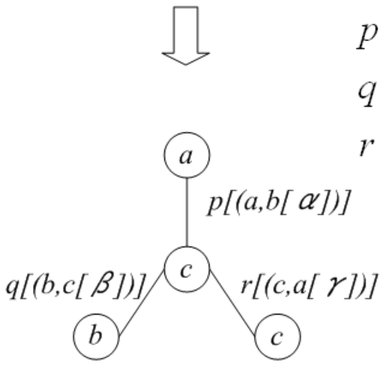


Figure 7: Serial and parallel transformations with path informations

While series and parallel transformations with path informations are trivial, path



$$\begin{aligned} u[\alpha] &= \min(x[l], y + z[a;m;b;n;c]) \\ v[\beta] &= \min(z[n], x + y[b;l;a;m;c]) \\ w[\gamma] &= \min(y[m], z + x[c;n;b;l;a]) \end{aligned}$$



$$\begin{aligned} p &= (u - v + w)/2 \\ q &= (u + v - w)/2 \\ r &= (-u + v + w)/2 \end{aligned}$$

Figure 8: $\Delta - Y$ transformations with path informations

information machinary of $\Delta - Y$ transformation is complicated but α, β, γ are variables for sequences of vertices which are determined by minimum operation. Note that after $\Delta - Y$ transformation complex path informations, e.g. $(a, b[\alpha])$, appear. This means the choice of path, e.g. a or $b[\alpha]$. Those complex path informations can be reduced by series transformation with the following rule. For instance '$\cdots; (a, b[\alpha])$' means that the suffix of sequence is $(a, b[\alpha])$.

l	m	$l; c; m$
$\cdots; (a, b[\alpha])$	$\cdots; (b, c[\beta])$	α
$\cdots; (b, c[\beta])$	$\cdots; (c, a[\gamma])$	β
$\cdots; (a, b[\alpha])$	$\cdots; (c, a[\gamma])$	γ

The following example (Figure 9) is an execution to search the shortest path and length from vertex s to vertex t. Every edge of initial graph does not have path informations.

5. Conclusion

Our method can be expanded to compute the shortest path not only the length of the shortest path. It is made by keeping information of removed vertices in labels of edges.

We showed the length preserving property of $\Delta - Y$ rules, this indicate a possibility for extending our method to an efficient algorithm for wider class of graphs

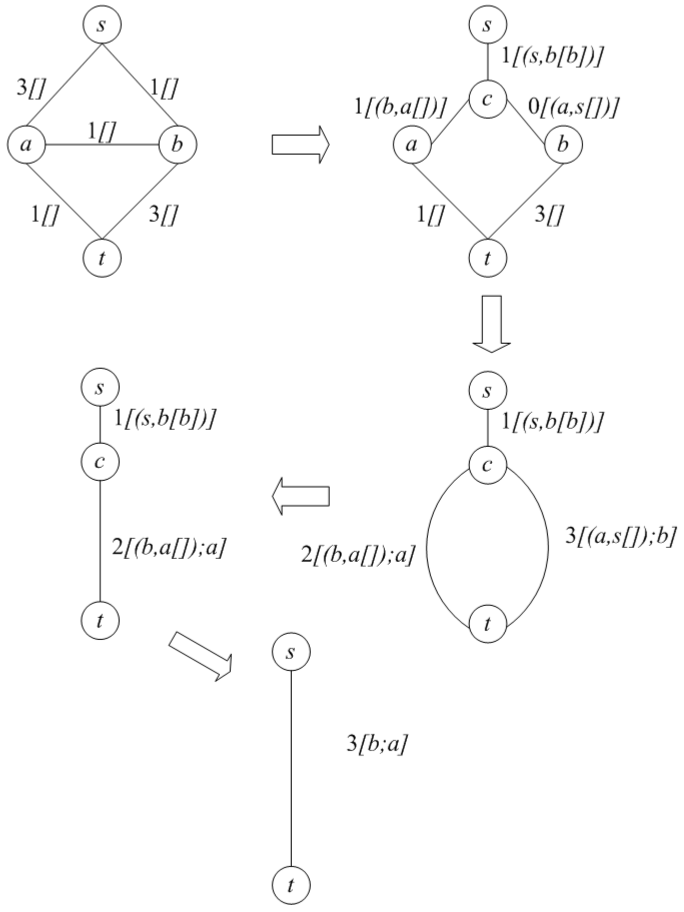


Figure 9: Example of searching the shortest path and length

than series parallel graphs. Especially, if results in Politof and Satyanarayana (1986b), Politof and Satyanarayana (1990) can be extended to compute 2-terminal reliability of networks, then it simply induce an $O(|E|)$ or $O(|V|)$ time algorithms for calculating the length of the shortest path.

References

Y. Okada and M. Hayashi. (1991). Graph rewriting systems and their application to network reliability analysis. *Lecture Notes in Computer Science*, Vol. 570, pp. 36–47, 1991.

T. Politof and A. Satyanarayana. (1986a). Efficient algorithms for reliability analysis of planar networks:A survey. *IEEE Transactions of reliability*, Vol. R-35, No. 3, pp.252–259, 1986.

T. Politof and A. Satyanarayana. (1986b). Network reliability and inner-four-cycle-free graphs. *Mathematics of operations research*, Vol. 11, No. 3, pp.484–505, 1986.

T. Politof and A. Satyanarayana. (1990). A liner time algorithm to compute the reliability of planar cube-free networks. *IEEE Transactions of reliability*, Vol. R-39, pp.557–563, 1990.

A. Satyanarayana and R.K. Wood. (1985). A liner-time algolithm for computing k-terminal reliability in series-parallel networks. *SIAM. J. Comput.*, Vol. 14, No. 4, pp. 818–832, 1985.

Y. Mizoguchi and Y. Kawahara (1995). Relational graph rewritings. *Theoretical Computer Science*, Vol. 141,pp.311-328,1995.

Y. Mizoguchi. (1999). Properties of graphs preserved by relational graph rewritings. *Information Science*, Vol. 119, pp.289–299, 1999.

Received 24th March 2004
Revised 26th March 2004