グラフデータベースにおける正則表現及び文脈自由文法を満たす最短経路の一探索法

鈴木伸崇[†] 佐藤洋一郎[†] 早瀬道芳[†]

半構造データ上では，その構造が不規則なので，与えられた経路式により適切な探索が行われるかどうかは必ずしも明確でない．本論文では，次のような理由から，経路式 pe により探索される経路の中で最短のもの，すなわち，pe を満たす最短経路を求めるアルゴリズムを考える: (i) pe を満たす経路を確認できれば，pe の働きが適切かどうかを検証できる．(ii) このような検証のためには，pe を満たす経路の中で（無駄な頂点や辺を迂回しない）より短い経路の方が適している．このアルゴリズムを，次の定式化の下で構成する: (i) 半構造データを重み付き有向グラフで表す．(ii) 経路式として正則表現及び文脈自由文法 (CFG) を用いる．R を正則表現，G_{cf} を CFG，m を正整数とする．本論文では，以下の 3 つの結果を示す．まず，すべての頂点対 (u,v) に対して u から v への R を満たす最短経路を求める多項式時間アルゴリズムを示す．次に，すべての頂点対 (u,v) に対して u から v への G_{cf} を満たす最短経路を求めるアルゴリズムを示す．最後に，すべての頂点対 (u,v) に対して (i) u から v への重みが m 以下の G_{cf} を満たす経路があるかどうかを決定し，(ii) もしそのような経路が存在するならば，u から v への G_{cf} を満たす最短経路を求めるアルゴリズムを示す．このアルゴリズムは，各辺の重みが 1 以上の場合，疑似多項式時間で動作する．

Finding Shortest Paths Satisfying Regular Expressions and Context-Free Grammars in Graph Databases

NOBUTAKA SUZUKI,[†] YOICHIROU SATO[†] and MICHIYOSHI HAYASE[†]

Over semistructured data, due to the irregularity it tends to be unclear if a given path expression retrieves appropriately. In this paper, we consider algorithms that find shortest paths retrieved by a given path expression pe (i.e. shortest paths satisfying pe) for the following reason: (i) whether pe retrieves appropriately can be verified by checking paths satisfying pe and (ii) for such a verification, paths containing less vertices/edges are more desirable. We construct the algorithms in the following context: (i) semistructured data is modeled by a weighted labeled directed graph and (ii) a path expression is represented by a regular expression or a context-free grammar (CFG). Let R be a regular expression, G_{cf} be a CFG, and m be a positive integer. In this paper, we present the following three results. First, we present a polynomial-time algorithm that finds a shortest path satisfying R from u to v for every pair of nodes u and v. Second, we present an algorithm that finds a shortest path satisfying G_{cf} from u to v for every pair of nodes u and v. Finally, we present an algorithm that decides for every pair of nodes u and v (i) whether there is a path p satisfying G_{cf} from u to v such that $w(p) \leq m$ and (ii) if such a path exists, then also finds a shortest path satisfying G_{cf} from u to v. The decision algorithm runs in pseudopolynomial time if every weight is positive.

1. はじめに

近年，半構造データに関する研究が盛んに行われている．半構造データは，通常，オブジェクトを頂点，オブジェクト間の参照関係を辺とする有向グラフとしてモデル化される[1),4)]．このようなデータ上での質問記述では，一般に，経路式を用いて航行条件を指定する．経路 p 上のラベル系列が経路式 pe の表す言語（ラベル系列の集合）に属するとき，p は pe を満たすという．本論文では，半構造データを重み付き有向グラフ，経路式を正則表現及び文脈自由文法 (CFG) とみなし，経路式を満たす経路を求めることを考える．ここで，経路式 pe とオブジェクト u,v に対して，u から v への pe を満たす経路は一般に複数存在するが，本論文ではこのような経路のうち最短経路を求めるアルゴリズムを示す．

半構造データでは，明確なスキーマ構造を有する従

† 岡山県立大学情報工学部
Department of Information and System Engineering, Okayama Prefectural University

来のデータと比べて，次のような理由から，経路式によりどのような探索が行われるのかを予測するのがより困難である．

- 明確なスキーマをもたず，正確なデータ構造が必ずしも明らかでない．
- データ構造が不規則なので，同じ種類のオブジェクト同士でも，(ある与えられたオブジェクトから) それらに至る経路のラベル系列は必ずしも一定でない．
- 「ラベル系列の集合」を表せる強力な経路式が用いられることが多く，その場合利用者の意図しない範囲まで探索が及ぶ可能性がある．

したがって，半構造データにおいては，経路式を満たす経路，すなわち，経路式で検索されるオブジェクトに至るまでの経路を確認し，その経路式の働きが適切かどうかを検証することが，従来のデータと比べてより重要となる．なお，前述のように経路式を満たす経路は一般に複数存在するが，上のような検証のためには，無駄な頂点や辺を迂回する冗長な経路より，できるだけ短い経路の方が適していると考えられる．そこで，本論文のアルゴリズムでは，経路式を満たす最短経路を求めている．

本論文では，データベースと経路式を次のように定式化する．まず，半構造データは一般に（必ずしも重みをもたない）有向グラフとして表されるが，本論文では,「最短経路」を自然に定式化できるという理由から，データベースを重み付き有向グラフと考える．もしグラフに適当な重みが定義されているならば（例えば，文献 7) では，各有向辺 $u \xrightarrow{l} v$ に対して「u と v の関連の強さ」に応じた重みを与えている），その重みを用いて，上述の「最短経路」は重み付き有向グラフ上での重み最小の経路として定式化できる．一方，重みをもたない通常のデータの場合，上述の「最短経路」は，辺数が最小の経路，すなわち，各辺の重みを 1 と仮定した上での重み最小の経路として定式化すればよい．次に，半構造データにおける経路式としてはワイルドカードや正則表現が現在最も広く使われており，近年では CFG に関する考察が一部で行われつつある [11]．よって，経路式に関して考察すべき文法は高々CFG までで十分と考えられるので，本論文では経路式として正則表現及び CFG を用いている．

以上の考察から，本論文では，半構造データ上での経路式を満たす最短経路を求める問題を，重み付き有向グラフにおける正則表現及び CFG に関する最短経路問題として定式化し，それを解くアルゴリズムを構成した．形式的には，R を正則表現または CFG，p を頂点 u から頂点 v への R を満たす経路とする．もし u から v への R を満たす任意の経路 p' に対して $w(p) \leq w(p')$ ならば，p は u から v への R を満たす最短経路であるという．ここで，$w(p)$ は p のすべての辺の重みの和を表す．R を正則表現，G_{cf} を CFG，m を正整数とする．本論文では，以下の 3 つの結果を示す．まず，すべての頂点対 (u, v) に対して u から v への R を満たす最短経路を求める多項式時間アルゴリズムを示す．次に，すべての頂点対 (u, v) に対して u から v への G_{cf} を満たす最短経路を求めるアルゴリズムを示す．最後に，すべての頂点対 (u, v) に対して (i) u から v への $w(p) \leq m$ かつ G_{cf} を満たす経路 p が存在するかどうかを決定し，(ii) もしそのような経路が存在するならば，u から v への G_{cf} を満たす最短経路を求めるアルゴリズムを示す．このアルゴリズムは，各辺の重みが 1 以上の場合，疑似多項式時間で動作する．

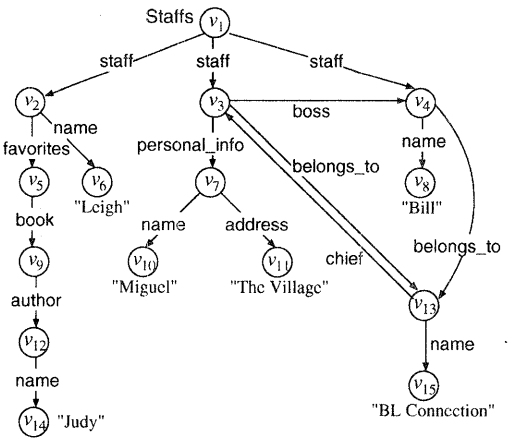


図 1 Staff データベース
Fig. 1 A staff database

質問中の経路式が，利用者の意図に沿った探索を行う適切なものかどうかが不明確なことがある．このような場合，経路式で探索される各頂点までの最短経路を列挙することで，その経路式が適切かどうかの検証を支援できる．例えば，図 1 の職員 (staff) に関するデータベースにおいて，すべての職員名を得たいとする．次の質問における経路式 $pe = \mathsf{staff.(\_)^*.name}$ が適切かどうか考える．ここで，'_' は任意のラベルを表す記号である．

```
select  A
from    Staffs.staff.(_)*.name  A
```

この質問は Lorel[2]（半構造データ用の質問言語）を模した構文で記述されており，from 文において頂点

表 1 staff.(_)*.name を満たす最短経路
Table 1 Shortest paths satisfying staff.(_)*.name

頂点	v_1 からの最短経路
v_6	$v_1 \xrightarrow{\text{staff}} v_2 \xrightarrow{\text{name}} v_6$
v_8	$v_1 \xrightarrow{\text{staff}} v_4 \xrightarrow{\text{name}} v_8$
v_{10}	$v_1 \xrightarrow{\text{staff}} v_3 \xrightarrow{\text{personal\_info}} v_7 \xrightarrow{\text{name}} v_{10}$
v_{14}	$v_1 \xrightarrow{\text{staff}} v_2 \xrightarrow{\text{favorites}} v_5 \xrightarrow{\text{book}} v_9 \xrightarrow{\text{author}} v_{12} \xrightarrow{\text{name}} v_{14}$
v_{15}	$v_1 \xrightarrow{\text{staff}} v_3 \xrightarrow{\text{belongs\_to}} v_{13} \xrightarrow{\text{name}} v_{15}$

v_1(Staffs) から pe により到達可能な各頂点が変数 A に代入され，select 文により A に代入された各頂点がこの質問の解となる．表 1 に，v_1 から pe により到達可能な頂点 $(v_6, v_8, v_{10}, v_{14}, v_{15})$，及び，$v_1$ からそれら各頂点への pe を満たす最短経路を列挙する（各辺の重みは 1 とする）．この表より，v_{14} と v_{15} へ至る経路は職員名とは無関係のラベル（book や belongs_to など）を含むので，この 2 頂点は職員名を表すものではないことが分かる．したがって，staff.(_)*.name は不要な頂点を検索する不適切な経路式である．

重み付きグラフ上で最短経路や正則表現を考慮したデータモデルはいくつか存在するが（文献 8) など），具体的なアルゴリズムは明示されていない．重み無しグラフにおいては，正則表現や CFG に関する形式的考察がいくつか行われている．例えば，文献 13) では正則表現と CFG を満たす経路の到達可能性判定問題について考察され，文献 12) では正則表現を満たす単純経路を求めるアルゴリズムが提案されている．他方，文献 10) ではオブジェクト指向データベースのスキーマグラフ上での探索が考察され，ワイルドカードを 1 つ含む経路式に対して，そのワイルドカードの部分を補完する経路を求めるアルゴリズムが提案されている．

2. 諸 定 義

Σ をアルファベットとする．Σ 上の正則表現 R を次のように再帰的に定義する．

- 空系列 ϵ，空集合 $\emptyset$，及び，$a \in \Sigma$.
- $(R+S)$, (RS)，及び，$(R)^*$．ここで，R と S は正則表現.

R で表される言語 $L(R)$ を次のように再帰的に定義する．

- $L(\epsilon) = \{\epsilon\}$，$L(\emptyset) = \emptyset$，かつ，各 $a \in \Sigma$ に対して $L(a) = \{a\}$.
- $L(R+S) = L(R) \cup L(S) = \{w \mid w \in L(R)$ または $w \in L(S)\}$.
- $L(RS) = L(R)L(S) = \{w_1 w_2 \mid w_1 \in L(R)$ かつ $w_2 \in L(S)\}$.
- $L(R^*) = \bigcup_{i=0}^{\infty} L(R)^i$．ここで，$L(R)^0 = \{\epsilon\}$ かつ $L(R)^i = L(R)^{i-1}L(R)$.

文脈自由文法 (CFG) は 4 項組 $G_{cf} = (V, \Sigma, P, S)$ で表され，ここで V は $V \cap \Sigma = \emptyset$ を満たす変数の集合，P は生成規則（後述）の集合，$S \in V$ は開始記号である．Σ に属する記号を終端記号と呼ぶ．P に属する各生成規則は $A \to \alpha$ という形であり，ここで $A \in V$ かつ $\alpha \in (V \cup \Sigma)^*$ である．任意の語 $\alpha, \gamma \in (V \cup \Sigma)^*$ に対して，もし $A \to \beta \in P$ ならば $\alpha A \gamma \Rightarrow \alpha\beta\gamma$ である．もし $\alpha_1 = \alpha$，$\alpha_n = \beta$，かつ，$1 \le i \le n-1$ に対して $\alpha_i \Rightarrow \alpha_{i+1}$ を満たす有限の系列 $\alpha_1, \alpha_2, \ldots, \alpha_n$ が存在するならば，α から β が導出されるといい，$\alpha \overset{*}{\Rightarrow} \beta$ と書く．$L(G_{cf}) = \{w \mid w \in \Sigma^*, S \overset{*}{\Rightarrow} w\}$ を G_{cf} によって生成される言語という．もし P に属するどの規則も $A \to BC$ または $A \to a$ という形であるならば，G_{cf} はチョムスキー標準形をしているという．ここで，$B, C \in V$ かつ $a \in \Sigma$ である．

$\mathbf{N}$ を非負整数全体の集合とする．データベースグラフ（グラフ）を 3 項組 $G = (N, E, w)$ と定義する．ここで，N は頂点の集合，E はラベル付き有向辺の集合，$w : E \to \mathbf{N}$ は重み関数である．どの $u, v \in N$ 及びどの $l \in \Sigma$ に対しても，G が u から v へのラベル l をもつ有向辺を複数もつことはないと仮定する．$p = (u, e_1, e_2, \ldots, e_l, v)$ を G における経路とする．ここで，各 $1 \le i \le l$ に対して $e_i = v_{i-1} \xrightarrow{a_i} v_i \in E$ かつ $v_i \in N$，$u = v_0$，かつ，$v = v_l$ である．p 上のラベル系列 $a_1 a_2 \ldots a_l$ を $\lambda(p)$ と表す．R を正則表現または CFG とする．もし $\lambda(p) \in L(R)$ ならば，p は R を満たすという．p の重みを $w(p) = \sum_{i=1}^{l} w(e_i)$ と定義する．u から v への経路 p を $u \overset{p}{\rightsquigarrow} v$ と表す．

3. では，ϵ-動作を含まない非決定性有限オートマトン (NFA) を用いる（議論の簡単のため，開始状態を複数とれるよう拡張されている）．形式的には，NFA は 5 項組 $M = (Q, \Sigma, \delta, S, F)$ で表され，ここで Q は状態の有限集合，$\delta : Q \times \Sigma \to 2^Q$ は遷移関数，$S \subseteq Q$ は初期状態の集合，$F \subseteq Q$ は最終状態の集合である．関数 $\hat{\delta}$ を $\hat{\delta}(q, \epsilon) = \{q\}$ かつ $\hat{\delta}(q, wa) = \bigcup_{q' \in \hat{\delta}(q,w)} \delta(q', a)$ と定義する．ここで，$q, q' \in Q$，$a \in \Sigma$，かつ，$w \in \Sigma^*$ である．もしある $q_s \in S$ に対して $\hat{\delta}(q_s, w) \cap F \neq \emptyset$ ならば，M は $w \in \Sigma^*$ を受理するという．M によって受理されるすべての語からなる集合を $L(M)$ と書く．M における状態 q から状態 q' への a による遷移を $q \xrightarrow[M]{a} q'$ と表す．ここで $q' \in \delta(q, a)$ である．

重み付き NFA(WNFA) は 6 項組 $W = (Q, \Sigma, \delta, S, F, w)$ で表され，ここで $(Q, \Sigma, \delta, S, F)$ は NFA，$w : \delta \to \mathbf{N}$ は各遷移に対して重みを割り当てる重み関数

である（関連するNFAが文献6)などにある）．遷移 $q \xrightarrow[W]{a} q'$ に割り当てられる重みを $w(q,a,q')$ と書く．$s = q_0 \xrightarrow[W]{a_1} q_1 \xrightarrow[W]{a_2} \cdots \xrightarrow[W]{a_n} q_n$ を遷移系列とする．s の重みを $w(s) = \sum_{i=1}^{n} w(q_{i-1}, a_i, q_i)$ と定義する．グラフ (Q, E, w') を W に関する遷移グラフという．ここで，$q_1 \xrightarrow{a} q_2 \in E \Leftrightarrow q_2 \in \delta(q_1, a)$，かつ，$w'$ は次のように定義される．

$$w'(q \xrightarrow{a} q') = \begin{cases} w(q,a,q') & q' \in \delta(q,a) \text{ のとき} \\ \infty & \text{それ以外のとき} \end{cases}$$

3. 正則表現を満たす最短経路

本章では，正則表現を満たす最短経路を求める多項式時間アルゴリズムを示す．$G = (N, E, w)$ をグラフ，$u, v \in N$ を頂点，R を正則表現とする．u から v への R を満たす経路の最小重み $d(u, v, R)$ を次のように定義する．

$$d(u,v,R) = \begin{cases} \min\{w(p) \mid u \overset{p}{\rightsquigarrow} v, \lambda(p) \in L(R)\} \\ \quad G \text{ が } R \text{ を満たす経路 } u \overset{p}{\rightsquigarrow} v \text{ を含むとき} \\ \infty \\ \quad \text{そうでないとき} \end{cases}$$

特に，$d(u, v, \Sigma^*)$ は正則表現を考慮しない通常の最短経路の重みを表し，以下その値を $d(u, v)$ と書くことにする．$w(p) = d(u, v, R)$ を満たす u から v への経路を u から v への R を満たす最短経路という．アルゴリズムの概要を以下に示す．

入力: グラフ $G = (N, E, w)$，Σ 上の正則表現 R
出力: すべての頂点対 (u, v) に対する u から v への R を満たす最短経路
段階 1: $L(M) = L(R)$ を満たすNFA M を構成する．
段階 2: G をWNFA W_G に変換する．
段階 3: $M \cap W_G$ を表すWNFA W_I を構成する．
段階 4: W_I に関する遷移グラフ上で（通常の）最短経路問題を解くことにより，各頂点対 (u, v) に対する u から v への R を満たす最短経路を求める．

段階1の詳細は省略する（文献3)を参照）．以下，段階2から段階4について述べる．

段階2: $G = (N, E, w)$ を表すWNFA W_G を構成する．G におけるすべての頂点を W_G の初期状態かつ最終状態とする．形式的には，$W_G = (N, \Sigma, \delta, N, N, w')$ と定義され，ここで $v_2 \in \delta(v_1, a) \Leftrightarrow v_1 \xrightarrow{a} v_2 \in E$，かつ，$w'$ は次のように定義される．

$$w'(v_1,a,v_2) = \begin{cases} w(v_1 \xrightarrow{a} v_2) & v_1 \xrightarrow{a} v_2 \in E \text{ のとき} \\ \infty & \text{そうでないとき} \end{cases}$$

段階3: $M = (Q, \Sigma, \delta, \{q_s\}, F)$ を段階1で得られるNFA，$W_G = (N, \Sigma, \delta', N, N, w')$ を段階2で得られるWNFAとする．本段階では $M \cap W_G$ を表すWNFA W_I を構成する．形式的には，$W_I = (Q \times N, \Sigma, \delta_I, \{q_s\} \times N, F \times N, w_I)$ と定義され，ここで δ_I と w_I は次のように定義される．

δ_I: 各 $a \in \Sigma$ に対して，$(q_2, v_2) \in \delta_I((q_1, v_1), a) \Leftrightarrow q_2 \in \delta(q_1, a)$ かつ $v_2 \in \delta'(v_1, a)$

w_I: $w_I((q_1, v_1), a, (q_2, v_2)) =$

$$\begin{cases} w'(v_1,a,v_2) & (q_2,v_2) \in \delta_I((q_1,v_1),a) \text{ のとき} \\ \infty & \text{そうでないとき} \end{cases}$$

次の補題が成り立つ．

補題1 $G = (N, E, w)$ をグラフ，$W_G = (N, \Sigma, \delta', N, N, w')$ を G を表すWNFA，R を正則表現，$M = (Q, \Sigma, \delta, \{q_s\}, F)$ を $L(M) = L(R)$ を満たすNFA，$W_I = (Q_I, \Sigma, \delta_I, S_I, F_I, w_I)$ を $M \cap W_G$ を表すWNFA，k を整数とする．このとき，以下の2条件は等しい．

S1: G は $\lambda(p) \in L(R)$ かつ $w(p) = k$ を満たす経路 $p = (u, e_1, e_2, \ldots, e_l, v)$ を含む．ここで，$u = u_0$，$v = u_l$，かつ，各 $1 \le i \le l$ に対して $e_i = u_{i-1} \xrightarrow{a_i} u_i \in E$ である．

S2: W_I において，$(q_0, u_0) = (q_s, u) \in S_I$，$(q_l, u_l) = (q_l, v) \in F_I$，かつ，$w(t) = k$ を満たす遷移系列 $t = (q_0, u_0) \xrightarrow[W_I]{a_1} (q_1, u_1) \xrightarrow[W_I]{a_2} \cdots \xrightarrow[W_I]{a_l} (q_l, u_l)$ が存在する．

（証明） 定義から，G が $u = u_0$，$v = u_l$，$e_i = u_{i-1} \xrightarrow{a_i} u_i$ $(1 \le i \le l)$，かつ，$w(p) = k$ を満たす経路 $p = (u, e_1, e_2, \ldots, e_l, v)$ を含むことと，W_G において $w(s) = k$ を満たす遷移系列 $s = u_0 \xrightarrow[W_G]{a_1} u_1 \xrightarrow[W_G]{a_2} \cdots \xrightarrow[W_G]{a_l} u_l$ が存在することは同値である．更に，$\lambda(p) \in L(R)$ であることと，M において $q_s = q_0$ かつ $q_l \in F$ を満たす遷移系列 $q_0 \xrightarrow[M]{a_1} q_1 \xrightarrow[M]{a_2} \cdots \xrightarrow[M]{a_l} q_l$ が存在することは同値である．したがって，条件S1は次の条件と等しい．

S3: M において $q_0 = q_s$ かつ $q_l \in F$ を満たす遷移系列 $q_0 \xrightarrow[M]{a_1} q_1 \xrightarrow[M]{a_2} \cdots \xrightarrow[M]{a_l} q_l$ が存在し，かつ，W_G において $u_0 = u$，$u_l = v$，かつ，$w(s) = k$ を満たす遷移系列 $s = u_0 \xrightarrow[W_G]{a_1} u_1 \xrightarrow[W_G]{a_2} \cdots \xrightarrow[W_G]{a_l} u_l$ が存在する．

$W_I = M \cap W_G$ なので，定義から条件S2と条件S3が等しいことは容易に示せる． □

段階4: $W_I = (Q_I, \Sigma, \delta_I, S_I, F_I, w_I)$ を段階3で得られた $M \cap W_G$ を表すWNFA，G_{W_I} を W_I に関する遷移グラフとする．定義から $S_I = \{q_s\} \times N$ であり，

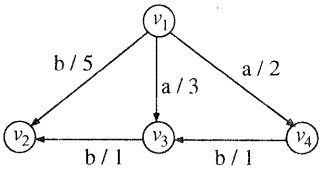


図 2 グラフ G
Fig. 2 A graph G

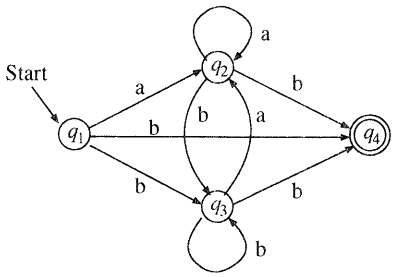


図 3 正則表現 $(a+b)^*b$ を表す NFA M
Fig. 3 An NFA M equivalent to regular expression $(a+b)^*b$

ここで q_s は M の初期状態，N は G の頂点集合である．u から v への R を満たす最短経路を求めるには，補題 1 より，G_{W_I} 上で $v'=v$ なる任意の $(q',v')\in F_I$ に対して $d((q_s,u),(q,v)) \le d((q_s,u),(q',v'))$ を満たす $(q,v)\in F_I$ を選択し，(q_s,u) から (q,v) への（通常の）最短経路を求めればよい．よって，以下の手順から，すべての頂点対 (u,v) に対する u から v への R を満たす最短経路を求めることができる．

(1) まず，G_{W_I} 上で，$(q_s,v_1)\in S_I$ かつ $(q,v_2)\in F_I$ を満たす各頂点対 $((q_s,v_1),(q,v_2))$ に対して $d((q_s,v_1),(q,v_2))$ の値を計算する．この値は，G_{W_I} に対して通常の最短経路アルゴリズム（Floyd-Warshall アルゴリズムなど）を適用すれば得られる．

(2) 次に，G における各頂点対 (u,v) に対して，G_{W_I} 上で $v'=v$ なる任意の $(q',v')\in F_I$ に対して $d((q_s,u),(q,v)) \le d((q_s,u),(q',v'))$ を満たす $(q,v)\in F_I$ を選択する．(q_s,u) から (q,v) への最短経路は，(1) の最短経路アルゴリズムの結果から得られる．

段階 1 は多項式時間で実行できる [3]．更に，他の 3 つの段階が多項式時間で実行できることも容易に示せる．したがって，次の定理が成り立つ．

定理 1 G をグラフ，R を正則表現とする．G において，すべての頂点対 (u,v) に対する u から v への R を満たす最短経路は多項式時間で求められる． □

例 1 $R=(a+b)^*b$ を正則表現とする．グラフ G

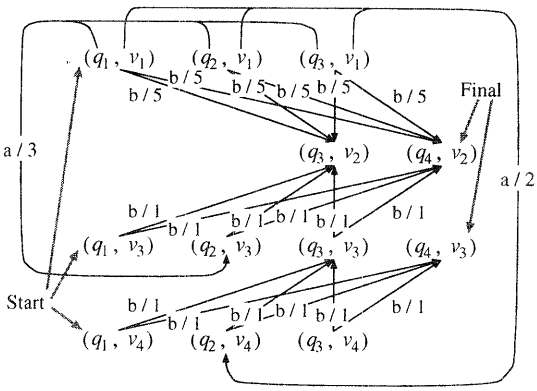


図 4 $M\cap W_G$ を表す WNFA W_I
Fig. 4 A WNFA W_I for $M\cap W_G$

（図 2）及び $L(M)=L(R)$ を満たす NFA M（図 3）を考える．M の初期状態は q_1，最終状態は q_4 である．以下，v_1 から v_2 への R を満たす最短経路 p を求めることを考える．W_G を G を表す WNFA とする．$M\cap W_G$ を表す WNFA W_I を図 4 に示す（孤立点は省略）．ここで，初期状態は (q_1,v_1)，(q_1,v_3)，及び，(q_1,v_4)，最終状態は (q_4,v_2) と (q_4,v_3) である．v_1 から v_2 への R を満たす最短経路を求めるためには，W_I に関する遷移グラフ（図 4）において (q_1,v_1) から (q_4,v_2) への最短経路を求めればよい．その最短経路は $(q_1,v_1)\overset{a/2}{\to}(q_2,v_4)\overset{b/1}{\to}(q_3,v_3)\overset{b/1}{\to}(q_4,v_2)$ なので，p は $v_1\overset{a/2}{\to}v_4\overset{b/1}{\to}v_3\overset{b/1}{\to}v_2$ となる． □

4. 文脈自由文法を満たす最短経路

本章では，CFG を満たす最短経路を求めることを考える．以下，一般性を失わず，各 CFG はチョムスキー標準形をしていると仮定する．

4.1 最短経路を求めるアルゴリズム

本節では，与えられた CFG を満たす最短経路を求めるアルゴリズムを示す．$G=(N,E,w)$ をグラフ，$u,v\in N$ を頂点，$G_{cf}=(V,\Sigma,P,S)$ を CFG，$A\in V$ を変数とする．$A\overset{*}{\Rightarrow}\lambda(p)$ を満たす経路 p を A-経路と呼ぶ．もし経路 p が S-経路ならば，p は G_{cf} を満たすという．G において u から v への A-経路が存在することを $u\rightsquigarrow_A v\in G$ と表す．u から v への A-経路の最小重み $d(u,v,A)$ を次のように定義する．

$$d(u,v,A)=\begin{cases}\min\{w(p)\mid u\overset{p}{\rightsquigarrow}v, A\overset{*}{\Rightarrow}\lambda(p)\} & u\rightsquigarrow_A v\in G \text{ のとき}\\ \infty & \text{そうでないとき}\end{cases}$$

$w(p) = d(u, v, A)$ を満たす u から v への A-経路 p を u から v への A-最短経路という．

以下に示すアルゴリズム 1 は，すべての頂点対 (u, v) に対して $d(u, v, S)$ を計算する（最短経路を表示するアルゴリズムは後述）．以下，n で G における頂点数を表す．アルゴリズム 1 は以下の行列を用いる．

(1) 各終端記号 $a \in \Sigma$ に対して，行列 D_a を用いる．$D_a[i, j]$ は辺 $v_i \xrightarrow{a} v_j \in E$ の重みを表す（アルゴリズムの 8〜14 行目）．

(2) 各変数 $A \in V$ に対して，行列 D_A を用いる．$D_A[i, j]$ は（その時点までに得られた最小の）v_i から v_j への A-経路の重みを保持する．

(3) 各変数 $A \in V$ に対して，行列 P_A を用いる．これは，最短経路の表示に用いられる（後述）．

アルゴリズム 1 の出力は S-経路の最小重みを保持する行列 D_S であり，$D_S[i, j] = d(v_i, v_j, S)$ $(1 \le i, j \le n)$ を満たす．1〜14 行目は行列の初期化である．15〜22 行目は，もし $A \to a \in P$ かつ $v_i \xrightarrow{a} v_j \in E$ ならば，v_i から v_j への A-経路で重みが $D_a[i, j] = w(v_i \xrightarrow{a} v_j)$ であるものが存在するという事実に対応する．23 行目の while 文は，どの行列の値も変化しなくなるまで繰り返される．26〜34 行目は，もしある $A \to BC \in P$ に対して，v_i から v_k への重み $D_B[i, k]$ の B-経路が存在し，かつ，v_k から v_j への重み $D_C[k, j]$ の C-経路が存在するならば，v_i から v_j への A-経路で重みが $D_A[i, j] = D_B[i, k] + D_C[k, j]$ であるものが存在するという事実に対応している．

アルゴリズム 1 CFG を満たす最短経路を計算
入力: グラフ $G = (N, E, w)$, CFG $G_{cf} = (V, \Sigma, P, S)$
出力: S-経路の最小重みを保持する行列 D_S

```
 begin
1.  for each A ∈ V do
2.    for i ← 1 to n do
3.      for j ← 1 to n do
4.        begin
5.          P_A[i,j] ← nil;
6.          D_A[i,j] ← ∞;
7.        end
8.  for each a ∈ Σ do
9.    for i ← 1 to n do
10.     for j ← 1 to n do
11.       if v_i -a-> v_j ∈ E then
12.         D_a[i,j] ← w(v_i -a-> v_j);
13.       else
14.         D_a[i,j] ← ∞;
15. for each A → a ∈ P do
16.   for i ← 1 to n do
17.     for j ← 1 to n do
18.       if D_A[i,j] > D_a[i,j] then
19.         begin
20.           P_A[i,j] ← ⟨a, 0⟩;
21.           D_A[i,j] ← D_a[i,j];
22.         end
23. while changes do
24.   for each A ∈ V do
25.     copy D_A to D_{A_old};
26.   for each A → BC ∈ P do
27.     for k ← 1 to n do
28.       for i ← 1 to n do
29.         for j ← 1 to n do
30.           if D_A[i,j] > D_{B_old}[i,k] + D_{C_old}[k,j] then
31.             begin
32.               P_A[i,j] ← ⟨BC, k⟩;
33.               D_A[i,j] ← D_{B_old}[i,k] + D_{C_old}[k,j];
34.             end
35. end /* while */
 end
```

以下，アルゴリズム 1 が次の条件を満たすことを示す．

P1: 任意の入力 (G, G_{cf}) に対して停止する．

P2: 任意の $1 \le i, j \le n$ に対して $D_S[i, j] = d(v_i, v_j, S)$ である．

まず，条件 P1 が成立することを示す．アルゴリズムの構成から，任意の $1 \le i, j \le n$ と任意の $A \in V$ に対して $D_A[i, j] \ge 0$ かつ $D_A[i, j] \le D_{A_{old}}[i, j]$ である．更に，23 行目の while 文が繰り返される間，$D_A[i, j] < D_{A_{old}}[i, j]$ を満たす $D_A[i, j]$ が少なくとも一つ存在する．したがって，条件 P1 が成り立つ．

次に，条件 P2 について，以下の 2 つの場合に分けて考える．

C1: $d(v_i, v_j, S) = \infty$

C2: $d(v_i, v_j, S) < \infty$

まず，C1 の場合を考える．以下に示す補題 2 より，もし G が v_i から v_j への S-経路を含まないならば，$D_S[i, j]$ の値は ∞ のまま更新されない．よって，$D_S[i, j] = d(v_i, v_j, S)$ であり条件 P2 が成り立つ．ここで，いくつかの記法を導入する．まず，次の時点における $D_A[i, j]$ の値を $D_A[i, j]^{(h,r,l)}$ と表す．

(1) while 文の繰り返し回数が h,

(2) 26 行目で選択された生成規則が r,

(3) 27 行目の k の値が l.

同様に，15 行目で生成規則 r が選択された時点での $D_A[i, j]$ の値を $D_A[i, j]^{(0,r,0)}$ と表す．次に，変数 $A \in V$ 及び A-経路 p に対して，A から $\lambda(p)$ への導出に対応する構文木を p に関する A-木と呼ぶことにする．例えば，$p = v_1 \xrightarrow{a} v_2 \xrightarrow{b} v_3 \xrightarrow{b} v_4$ を経路，$G_{cf} = (V, \Sigma, P, S)$ を CFG とする．ここで，$V = \{S, A, B\}$, $\Sigma = \{a, b\}$, かつ，$P = \{S \to BA, A \to AB, A \to a, B \to b\}$ である．このとき，図 5 の構文木は p に関する高さ 3 の A-木であり，導出 $A \Rightarrow AB \Rightarrow ABB \Rightarrow aBB \Rightarrow abB \Rightarrow abb$ に対応する．(構文木の詳細は文献 9) を参照.)

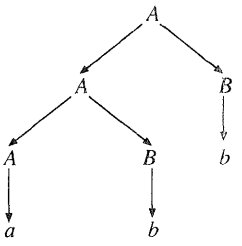


図 5 p に関する A-木
Fig. 5 An A-tree for p

補題 2 $G=(N,E,w)$ をグラフ，(V,Σ,P,S) をCFG，$A\in V$ を変数とする．アルゴリズム 1 の実行において，もしある $h\geq 0$，ある $r\in P$，及び，ある $0\leq k\leq l-1$ に対して $D_A[i,j]^{(h,r,k)}$ の値が x $(0\leq x<\infty)$ に更新されたならば，G は次の 2 条件を満たすA-経路 $p=(v_i,e_1,\ldots,e_l,v_j)$ を含む．

(1) $w(p)=x$ である．

(2) p に関する，高さ $h+1$ のA-木が存在する．

（証明） h に関する帰納法で示す．

初期段: $h=0$ である．$r=A\to a\in A$ に対して $D_A[i,j]^{(0,r,0)}$ の値が 21 行目で $x=D_a[i,j]$ に更新されたと仮定する．$x=D_a[i,j]$ なので，8～14 行目より $x=D_a[i,j]=w(v_i\xrightarrow{a}v_j)$ を満たす辺 $v_i\xrightarrow{a}v_j\in E$ が存在する．よって，G は $w(p)=w(e_1)=x$ なるA-経路 $p=(v_i,e_1,v_j)$ を含み，ここで $e_1=v_i\xrightarrow{a}v_j$ である．したがって条件 (1) が成り立つ．また，$A\to a\in P$ かつ $e_1=v_i\xrightarrow{a}v_j$ なので，構文木の定義から条件 (2) が成り立つ．

帰納段: 帰納法の仮定として，もしある $h<m$ $(m\geq 1)$，ある $r\in P$，及び，ある $1\leq k\leq l-1$ に対して $D_A[i,j]^{(h,r,k)}$ の値が x に更新されたならば，条件 (1) と (2) を満たすA-経路 $p=(v_i,e_1,\ldots,e_l,v_j)$ が存在すると仮定する．$h=m$ の場合を考える．ある $r=A\to BC\in P$ 及びある $1\leq k\leq l-1$ に対して，(33 行目において) $D_A[i,j]^{(h,r,k)}$ が $x=D_{B_{old}}[i,k]^{(h,r,k)}+D_{C_{old}}[k,j]^{(h,r,k)}$ に更新されたとする．まず，条件 (1) を満たすA-経路が存在することを示す．$b=D_{B_{old}}[i,k]^{(h,r,k)}$ かつ $c=D_{C_{old}}[k,j]^{(h,r,k)}$ とする．$D_A[i,j]^{(h,r,k)}$ の値が $x=b+c$ に更新されたので，30 行目から $b<\infty$ かつ $c<\infty$ である．したがって，(i) ある $h_1<m$，ある $r_1\in P$，及び，ある $1\leq k_1\leq l-1$ に対して $D_B[i,k]^{(h_1,r_1,k_1)}$ の値が b に更新され，かつ，(ii) ある $h_2<m$，ある $r_2\in P$，及び，ある $1\leq k_2\leq l-1$ に対して $D_C[k,j]^{(h_2,r_2,k_2)}$ の値が c に更新されている．$h_1<m$ かつ $D_B[i,k]^{(h_1,r_1,k_1)}$ は b に更新されるので，帰納法の仮定から G は $w(p_1)=b$ かつ高さ h_1+1 のB-木をもつB-経路 $p_1=(v_i,e_1,\ldots,e_{l_1},v_k)$ を含む．同様に，$D_C[k,j]^{(h_2,r_2,k_2)}$ は c に更新されかつ $h_2<m$ なので，G は $w(p_2)=c$ かつ高さ h_2+1 のC-木をもつC-経路 $p_2=(v_k,e_{l_2},\ldots,e_l,v_j)$ を含む．$A\to BC\in P$，p_1 はB-経路，かつ，p_2 はC-経路なので，$A\Rightarrow BC\overset{*}{\Rightarrow}\lambda(p_1)\lambda(p_2)$ である．p_1 と p_2 を接続して得られる経路 $p=(v_i,e_1,\ldots,e_{l_1},v_k,e_{l_2},\ldots,e_l,v_j)$ は，$A\overset{*}{\Rightarrow}\lambda(p_1)\lambda(p_2)=\lambda(p)$ かつ $D_A[i,j]^{(h,r,k)}=b+c=w(p_1)+w(p_2)=w(p)$ を満たし，ここで $b+c=x$ である．したがって，条件 (1) が成り立つ．次に，p が条件 (2) を満たすことを示す．$D_A[i,j]^{(h,r,k)}$ は while 文の h 回目の繰り返しにおいて更新されるので，$D_B[i,k]^{(h_1,r_1,k_1)}$ と $D_C[k,j]^{(h_2,r_2,k_2)}$ のうち少なくとも一方は $(h-1)$ 回目に更新されている．よって，$\max(h_1,h_2)=h-1$ である．ここで，A が根，その左部分木が上述の p_1 に関するB-木，右部分木が p_2 に関するC-木であるA-木 t を考える．$\lambda(p_1)\lambda(p_2)=\lambda(p)$ より t は p に関するA-木であり，その高さは $\max(h_1+1,h_2+1)+1=h+1$ である．したがって，p は条件 (2) を満たす． □

以下，条件 P2 が C2 の場合において成り立つことを示す．そのため，いくつかの補題と定義を与える．まず，次の補題が成り立つ（証明は付録 A.1）．

補題 3 $G=(N,E,w)$ をグラフ，(V,Σ,P,S) をCFG，$A\in V$ を変数，$p=(v_i,e_1,\ldots,e_l,v_j)$ を G における A-最短経路とする．もしある $A\to BC\in P$ に対して p をB-経路 $p_1=(v_i,e_1,\ldots,e_{l_1},v_k)$ とC-経路 $p_2=(v_k,e_{l_2},\ldots,e_l,v_j)$ に分割できるならば，$w(p_1)=d(v_i,v_k,B)$ かつ $w(p_2)=d(v_k,v_j,C)$ である． □

次に，A-木の高さに関する記法を導入する．A-経路 p に関するA-木の最小の高さを $h(p,A)=\min\{h(t)\mid t$ は p に関するA-木 $\}$ と定義する．ここで，$h(t)$ は木 t の高さである．$v_i\rightsquigarrow_A v_j\in G$ を満たす頂点 v_i,v_j に対して，v_i から v_j への A-最短経路は複数存在することがある．v_i から v_j へのすべての A-最短経路に関するA-木の高さのうち，最小の値を $h(v_i,v_j,A)$ と表す．すなわち，

$$h(v_i,v_j,A)=\begin{cases}\min\{h(p,A)\mid v_i\overset{p}{\rightsquigarrow}v_j, w(p)=d(v_i,v_j,A), & \\ \quad A\overset{*}{\Rightarrow}\lambda(p)\} & v_i\rightsquigarrow_A v_j\in G \text{ のとき}\\ \infty & \text{そうでないとき}\end{cases}$$

次の補題が成り立つ（証明は付録 A.2）．

補題 4 $G=(N,E,w)$ をグラフ，$v_i,v_j\in N$ を頂

点，(V, Σ, P, S) を CFG，$A \in V$ を変数とする．もし $2 \leq h(v_i, v_j, A) < \infty$ ならば，ある頂点 v_k とある生成規則 $A \to BC \in P$ に対して $h(v_i, v_j, A) = \max(h(v_i, v_k, B), h(v_k, v_j, C)) + 1$ である． □

ここで，次の補題を示す．この結果から，もし $v_i \rightsquigarrow_A v_j \in G$ ならば $D_A[i,j] = d(v_i, v_j, A)$ であり，C2 の場合において条件 P2 が成立する．

補題 5 $G = (N, E, w)$ をグラフ，$v_i, v_j \in N$ を頂点，(V, Σ, P, S) を CFG，$A \in V$ を変数とする．もし $v_i \rightsquigarrow_A v_j \in G$ ならば，(i) アルゴリズム 1 の while 文の $(h(v_i, v_j, A) - 1)$ 回目の繰り返しにおいて $D_A[i,j]$ の値は $d(v_i, v_j, A)$ に更新され，かつ，(ii) その値はそれ以降更新されない．

（証明） まず (ii) を示す．補題 2 より，任意の $1 \leq i, j \leq n$，$h \geq 0$，$r \in P$，及び，$0 \leq k \leq l-1$ に対して $D_A[i,j]^{(h,r,k)} \geq d(v_i, v_j, A)$ である．したがって，$D_A[i,j]$ の値が $d(v_i, v_j, A)$ に更新された場合，30 行目の条件からその値はそれ以降更新されない．

次に，(i) が成り立つことを $h(v_i, v_j, A)$ に関する帰納法で示す．

初期段: $h(v_i, v_j, A) = 1$ である．定義から，ある $a \in \Sigma$ に対して $w(v_i \xrightarrow{a} v_j) = d(v_i, v_j, A)$ を満たす生成規則 $A \to a \in P$ 及び辺 $v_i \xrightarrow{a} v_j \in E$ が存在する．このとき，21 行目から $D_A[i,j] = w(v_i \xrightarrow{a} v_j) = d(v_i, v_j, A)$ である．

帰納段: 帰納法の仮定として，もし $v_i \rightsquigarrow_A v_j \in G$ かつ $h(v_i, v_j, A) < h$ $(h \geq 2)$ ならば，while 文の $(h(v_i, v_j, A) - 1)$ 回目の繰り返しにおいて $D_A[i,j]$ の値が $d(v_i, v_j, A)$ に更新されると仮定する．$h(v_i, v_j, A) = h$ の場合を考える．補題 4 から，次の式を満たす頂点 v_k 及び生成規則 $A \to BC \in P$ が存在する．

$$\max(h(v_i, v_k, B), h(v_k, v_j, C)) = h - 1 \quad (1)$$

式 (1) より $h(v_i, v_k, B) < h$ なので，帰納法の仮定から while 文の $(h(v_i, v_k, B) - 1)$ 回目の繰り返しにおいて $D_B[i,k]$ の値が $d(v_i, v_k, B)$ に更新され，(ii) よりその値はそれ以降更新されない．同様に，while 文の $(h(v_k, v_j, C) - 1)$ 回目の繰り返しにおいて $D_C[k,j]$ の値が $d(v_k, v_j, C)$ に更新され，以降その値は更新されない．式 (1) と帰納法の仮定から，$D_B[i,k]$ と $D_C[k,j]$ の少なくとも一方は while 文の $(h-2)$ 回目の繰り返しにおいてその値が更新される．このとき，補題 3 より $D_B[i,k] + D_C[k,j] = d(v_i, v_k, B) + d(v_k, v_j, C) = d(v_i, v_j, A)$ である．よって，while 文の $(h-1)$ 回目の繰り返しにおいて $D_A[i,j]$ の値が $d(v_i, v_j, A)$ に更新されることを示すには，$(h-2)$ 回目の繰り返しまでは $D_A[i,j] > d(v_i, v_j, A)$ であることを示せばよい．

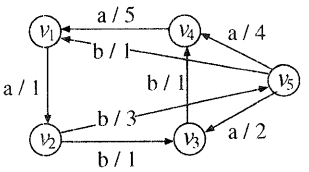


図 6 グラフ G
Fig. 6 A graph G

表 2 行列 D_a と D_b の値
Table 2 The values of matrices D_a and D_b

$$D_a = \begin{pmatrix} \infty & 1 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 5 & \infty & \infty & \infty & \infty \\ \infty & \infty & 2 & 4 & \infty \end{pmatrix}$$

$$D_b = \begin{pmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 1 & \infty & 3 \\ \infty & \infty & \infty & 1 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 1 & \infty & \infty & \infty & \infty \end{pmatrix}$$

$h(v_i, v_j, A) = h$ なので，定義より v_i から v_j への A-最短経路で高さ h 未満の A-木をもつようなものは存在しない．したがって，補題 2 より while 文の繰り返しが $(h-2)$ 回目に至るまでは $D_A[i,j] > d(v_i, v_j, A)$ である． □

アルゴリズム 1 の時間計算量を求めることは今後の課題であるが，すべての辺の重みを 0 に限定した場合，アルゴリズム 1 は多項式時間で動作する（各 $D_S[i,j]$ は 0 か ∞ のいずれか一方の値しか取らないことに注意）．この場合，D_S は任意の $1 \leq i, j \leq n$ に対して次の式 (2) を満たす，すなわち，アルゴリズム 1 は CFG に関する到達可能性（G が v_i から v_j への G_{cf} を満たす経路を含むかどうか）を決定する．

$$D_S[i,j] = \begin{cases} 0 & v_i \rightsquigarrow_S v_j \in G \text{ のとき} \\ \infty & \text{そうでないとき} \end{cases} \quad (2)$$

次の定理が成り立つ．

定理 2 $G = (N, E, w)$ をグラフ，G_{cf} を CFG とする．すべての頂点対 (u, v) に対して u から v への G_{cf} を満たす最短経路を求める問題は可解である．特に，もし各 $e \in E$ に対して $w(e) = 0$ ならば，同問題は多項式時間可解である． □

例 2 $G = (N, E, w)$ を図 6 に示されるグラフ，$G_{cf} = (V, \Sigma, P, S)$ を CFG とする．ここで，$V = \{S, A, B\}$，$\Sigma = \{a, b\}$，かつ，$P = \{S \to BA, A \to AB, A \to a, B \to b\}$ である．入力 G, G_{cf} に対してアルゴリズム 1 を実行する．8〜14 行目において D_a と D_b が初期化される（表 2）．While 文の各繰り返しにおける D_A，D_B，及び，D_S の値を表 3 に示す．まず，0 回目の繰り返し（15〜22 行目）を考

表 3 while 文の各繰り返しにおける D_A, D_B, 及び, D_S の値
Table 3 The values of D_A, D_B, and D_S for each iteration of the while loop

繰り返し回数	D_A	D_B	D_S
0	$\begin{pmatrix} \infty & 1 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 5 & \infty & \infty & \infty & \infty \\ \infty & \infty & 2 & 4 & \infty \end{pmatrix}$	$\begin{pmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 1 & \infty & 3 \\ \infty & \infty & \infty & 1 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 1 & \infty & \infty & \infty & \infty \end{pmatrix}$	$\begin{pmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{pmatrix}$
1	$\begin{pmatrix} \infty & 1 & 2 & \infty & 4 \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 5 & \infty & \infty & \infty & \infty \\ \infty & \infty & 2 & 3 & \infty \end{pmatrix}$	$\begin{pmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 1 & \infty & 3 \\ \infty & \infty & \infty & 1 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 1 & \infty & \infty & \infty & \infty \end{pmatrix}$	$\begin{pmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 5 & 7 & \infty \\ 6 & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & 2 & \infty & \infty & \infty \end{pmatrix}$
2	$\begin{pmatrix} 5 & 1 & 2 & 3 & 4 \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 5 & \infty & \infty & \infty & \infty \\ \infty & \infty & 2 & 3 & \infty \end{pmatrix}$	$\begin{pmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 1 & \infty & 3 \\ \infty & \infty & \infty & 1 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 1 & \infty & \infty & \infty & \infty \end{pmatrix}$	$\begin{pmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 5 & 6 & \infty \\ 6 & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & 2 & 3 & \infty & 5 \end{pmatrix}$
3 (及び 4)	$\begin{pmatrix} 5 & 1 & 2 & 3 & 4 \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 5 & \infty & \infty & \infty & \infty \\ \infty & \infty & 2 & 3 & \infty \end{pmatrix}$	$\begin{pmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 1 & \infty & 3 \\ \infty & \infty & \infty & 1 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 1 & \infty & \infty & \infty & \infty \end{pmatrix}$	$\begin{pmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 5 & 6 & \infty \\ 6 & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 6 & 2 & 3 & 4 & 5 \end{pmatrix}$

える．$A \to a, B \to b \in P$ なので，D_A（D_B）の値は D_a（D_b）のそれと等しい．次に，1 回目の繰り返し（23〜35 行目）を考える．$A \to AB \in P$ かつ $D_A[3,4] = 4 > D_{A_{old}}[5,3] + D_{B_{old}}[3,4] = 2 + 1 = 3$ なので，33 行目において $D_A[3,4]$ の値は 3 に更新される．他の成分も同様にして更新される．While 文の 2 回目以降の繰り返しも同様に実行され，4 回目の繰り返しではどの行列も更新されず while 文は停止する．$D_S[i,j]$ は v_i から v_j への S-経路の最小重みを表す．例えば，$D_S[5,1] = 6$ は G が v_5 から v_1 への重み 6 の S-最短経路 p を含むことを表す．次節の最短経路を表示するアルゴリズムを用いると，$p = v_5 \xrightarrow{b/1} v_1 \xrightarrow{a/1} v_2 \xrightarrow{b/3} v_5 \xrightarrow{b/1} v_1$ を得る． □

最短経路を表示するアルゴリズム

本節では，与えられた頂点対 (v_i, v_j) に対して，v_i から v_j への S-最短経路を表示するアルゴリズムを示す．これは，アルゴリズム 1 の行列 P_A $(A \in V)$ を用いて最短経路を表示する．アルゴリズム 1 の構成から，行列 P_A が任意の $1 \le i, j \le n$ に対して次の条件を満たすことは容易に示せる．

- もし $P_A[i,j] = \langle BC, k \rangle$ ならば，$v_i \rightsquigarrow_A v_j \in G$, $A \to BC \in P$, $D_A[i,j] = d(v_i, v_j, A)$, $D_B[i,k] = d(v_i, v_k, B)$, かつ，$D_C[k,j] = d(v_k, v_j, C)$ である．
- もし $P_A[i,j] = \langle a, 0 \rangle$ ならば，$v_i \xrightarrow{a} v_j \in E$, $A \to a \in P$, かつ，$D_A[i,j] = d(v_i, v_j, A) = w(v_i \xrightarrow{a} v_j)$ である．
- もし $P_A[i,j] = nil$ ならば，G は v_i から v_j への A-経路を含まない．

したがって，v_i から v_j への S-最短経路を表示するアルゴリズムは次のように構成できる．

アルゴリズム 2 最短経路の表示
入力: 頂点 v_i, v_j, 行列集合 $\{P_A \mid A \in V\}$
出力: v_i から v_j への最短経路

```
begin
1.  if P_S[i,j] = nil then exit;
2.  write("v_i");
3.  path(S,i,j); /* 下で定義されるサブルーチン */
4.  write("v_j");

5.  subroutine path(A,i,j)
6.  begin
7.    if P_A[i,j] = ⟨BC,k⟩ then
8.      begin
9.        path(B,i,k);
10.       write("v_k");
11.       path(C,k,j);
12.     end
13.   else if P_A[i,j] = ⟨a,0⟩ then
14.     write("--a-->");
15. end /* path */
end
```

4.2 決定アルゴリズム

$G = (N, E, w)$ をグラフ，$G_{cf} = (V, \Sigma, P, S)$ を CFG，m を正整数とする．以下，次の条件が成り立つと仮定する．

$$\text{任意の } e \in E \text{ に対して } w(e) \ge 1 \qquad (3)$$

本節では，すべての頂点対 (u, v) に対して以下を計算する疑似多項式時間アルゴリズムを示す．

(1) $d(u, v, S) \le m$ かどうかを決定する．
(2) もし $d(u, v, S) \le m$ ならば，u から v への G_{cf} を満たす最短経路を求める．

このアルゴリズムは，(i) 入力として m が追加され，(ii) 23 行目の while 文が次の repeat 文に置き換えられる，という 2 点を除いてアルゴリズム 1 と同じである．ここで，$d = \min\{w(e) \mid e \in E\}$ である．

23. **repeat** $\lceil \frac{m}{d} \rceil - 1$ **times**

すなわち，このアルゴリズムは24～34行目を（定常状態まで繰り返すのではなく）$(\lceil \frac{m}{d} \rceil - 1)$ 回だけ繰り返す．

同アルゴリズムの正当性を示す．(i) $d(v_i, v_j, S) \leq m$ のとき $D_S[i,j] = d(v_i, v_j, S)$，(ii) そうでないとき $D_S[i,j] > m$ であることを示せばよい．以下 (i) のみを示すが，(ii) も同様にして示せる．$d(v_i, v_j, S) \leq m$ と仮定する．経路 p のもつ辺の個数を $len(p)$ と表す．構文木の定義から，S-経路 p に関する S-木の高さは高々 $len(p)$ である．したがって，v_i から v_j への任意の S-最短経路 p は $h(v_i, v_j, S) \leq len(p)$ を満たす．更に，式 (3) より任意の経路 p に対して $d \cdot len(p) \leq w(p)$ である．$h(v_i, v_j, S) \leq len(p)$ かつ $d \cdot len(p) \leq w(p)$ なので，v_i から v_j への任意の S-最短経路 p は $d \cdot h(v_i, v_j, S) \leq w(p)$ を満たす．したがって，定義から $d \cdot h(v_i, v_j, S) \leq d(v_i, v_j, S)$ である．$d \cdot h(v_i, v_j, S) \leq d(v_i, v_j, S)$ と $d(v_i, v_j, S) \leq m$ より $d \cdot h(v_i, v_j, S) \leq m$ であり，よって $h(v_i, v_j, S) \leq \frac{m}{d} \leq \lceil \frac{m}{d} \rceil$ である．$h(v_i, v_j, S) - 1 \leq \lceil \frac{m}{d} \rceil - 1$ かつ repeat 文は $\lceil \frac{m}{d} \rceil - 1$ 回繰り返されるので，補題5より $D_S[i,j] = d(v_i, v_j, S)$ である．

このアルゴリズムの時間計算量が $O((|V| + |\Sigma|) \cdot n^2 + \frac{m}{d} \cdot |P| \cdot n^3)$ であることは容易に示せる．この計算量は（d を固定すると）m に関して疑似多項式である．したがって，m が著しく大きくない限り，このアルゴリズムは効率良く動作する．次の定理が成り立つ．

定理3 $G = (N, E, w)$ を任意の $e \in E$ に対して $w(e) \geq 1$ を満たすグラフ，G_{cf} を CFG，m を正整数とする．このとき，すべての頂点対 (u, v) に対して以下を計算する問題は疑似多項式時間可解である．

(1) $d(u, v, S) \leq m$ かどうかを決定する．

(2) もし $d(u, v, S) \leq m$ ならば，u から v への G_{cf} を満たす最短経路を求める．

□

例3 G と G_{cf} をそれぞれ例2におけるグラフと CFG とする．本節のアルゴリズムを G，G_{cf}，及び，$m = 3$ に対して実行することを考える．$d = \min\{w(e) \mid e \in E\} = 1$ かつ $\lceil \frac{m}{d} \rceil = 3$ なので，repeat 文は $\frac{3}{1} - 1 = 2$ 回繰り返され，よって同アルゴリズムの出力は表3における3行目の D_S の値と一致する．例えば，$D_S[5,3] = 3$ は，G が v_5 から v_3 への重み3の S-最短経路 p を含むことを表す．アルゴリズム2より $p = v_5 \xrightarrow{b/1} v_1 \xrightarrow{a/1} v_2 \xrightarrow{b/1} v_3$ である．一方，$D_S[2,4] = 6 > 3$ なので，G は v_2 から v_4 への重みが3以下の S-経路を含まない． □

5. むすび

本論文では，次のような定式化の下で，半構造データにおける経路式を満たす最短経路を求めるアルゴリズムを示した．

- データベースを，重み付き有向グラフで表す．
- 経路式として，正則表現及び CFG を用いる．

今後の課題としては，アルゴリズム1のより厳密な時間計算量を求めることが挙げられる．

CFG は正則表現より強力である反面，航行条件の記述がより複雑となる．今後は，正則表現や CFG 以外の文法で，経路式の記述が容易かつ最短経路が効率よく求まるものがあるかどうか調査する予定である．

1. での最短経路の列挙による経路式の検証においては，最短経路だけでなく，第 k 最短経路（重みの小さいものから順に k 番目の経路）まで順に列挙できればより有用であると考えられる．正則表現を満たす第 k 最短経路を求めるには，既存の第 k 最短経路アルゴリズム（文献5）など）を W_I に関する遷移グラフ（**3.** の段階4）に適用すればよい．他方，CFG を満たす第 k 最短経路を求める効率のよいアルゴリズムについて考察することは今後の課題である．

謝辞 貴重な御意見を頂いた査読者の方々に深謝致します．また，種々御協力頂いた奈良先端科学技術大学院大学権娟大氏に深謝致します．

参考文献

1) Abiteboul, S.: Querying Semi-Structured Data, *Proc. ICDT*, LNCS 1186, pp. 1–18, (1997).
2) Abiteboul, S., Quass, D., McHugh, J., Widom, J. and Wiener, L.: The Lorel Query Language for Semistructured Data, *Journal of Digital Libraries*, Vol. 1, No. 1, pp. 68–88, (1997).
3) Aho, A. V., Hopcroft, J. E. and Ullman, J. D.: *The Design and Analysis of Computer Algorithms*, Addsion-Wesly, Reading, MA, (1974).
4) Buneman, P.: Semistructured Data, *Proc. PODS*, pp. 117–121, (1997).
5) Chong, E., Maddila, S. and Morley, S.: On Finding Single-Source Single-Destination k Shortest Paths, *Proc. 7th Int. Conf. Computing and Information*, pp. 40–47, (1995).
6) Derencourt, D., Karhumaki, J., Latteux, M. and Terlutte, A.: On Computational Power of Weighted Finite Automata, *Proc. Int. Symp. Mathematical Foundations of Computer Science*, LNCS 629, pp. 236-245, (1992).

7) Goldman, R., Shivakumar, N., Venkatasubramanian, S. and Gracia-Molina, H.: Proximity Search in Databases, *Proc. VLDB*, New York, pp. 26–37, (1998).
8) Güting, R.: GraphDB: Modeling and Querying Graphs in Databases, *Proc. VLDB*, Santiago, Chile, pp. 297–308, (1994).
9) Hopcroft, J. E. and Ullman, J. D.: *Introduction to Automata Theory, Languages and Computation*, Addsion-Wesly, Reading, MA, (1979).
10) Ioannidis, Y. E. and Lashkari, Y.: Incomplete Path Expressions and Their Disambiguation, *Proc. ACM SIGMOD Conf.*, Minneapolis, Minnesota, pp. 138–149, (1994).
11) Ludascher, B., Papakonstantinou, Y., Velikhov, P., and Vianu, V.: View Definition and DTD Inference for XML, *Proc. Workshop on Query Processing for Semistructured Data and Non-Standard Data Formats (in conjunction with ICDT'99)*, (1999).
12) Mendelzon, A. O. and Wood, P. T.: Finding Regular Simple Paths in Graph Databases, *SIAM J. Comput.*, Vol. 24, No. 6, pp.1235–1258, (1995).
13) Yannakakis, M.: Graph-Theoretic Methods in Database Theory, *Proc. PODS*, Nashville, Tennessee, pp. 230–242, (1990).
14) 吉川正俊, 田中克己, 上善恒雄, 田中康暁, 蛭井潤, 堀田光治郎: ObaseLang:柔軟な構文と拡張経路式を持つオブジェクトデータベース言語, 情処学論, Vol. 36, No. 4, pp. 981–993, (1995).

付　　録

A.1 補題 3 の証明

$p = (v_i, e_1, \ldots, e_l, v_j)$ を G における A-最短経路，かつ，ある生成規則 $A \to BC \in P$ に対して p はB-経路 $p_1 = (v_i, e_1, \ldots, e_{l_1}, v_k)$ とC-経路 $p_2 = (v_k, e_{l_2}, \ldots, e_l, v_j)$ に分割できると仮定する．以下，背理法で示す．$w(p_1) > d(v_i, v_k, B)$ と仮定する（$w(p_2) > d(v_k, v_j, C)$ の場合も同様）．$w(p_1) > d(v_i, v_k, B)$ なので，p_1 と異なるB-経路 $p_1' = (v_i, e_1', \ldots, e_{l'}', v_k)$ で $w(p_1') = d(v_i, v_j, B)$ を満たすものが存在する．$p' = (v_i, e_1', \ldots, e_{l'}', v_k, e_{l_2}, \ldots, e_l, v_j)$ を p_1' と p_2 を接続して得られる経路とする．$A \to BC \in P$, $B \stackrel{*}{\Rightarrow} \lambda(p_1')$, かつ，$C \stackrel{*}{\Rightarrow} \lambda(p_2)$ なので，p' は v_i から v_j へのA-経路である．更に，$w(p') = w(p_1') + w(p_2) < w(p_1) + w(p_2) = w(p)$ である．しかし，$w(p') < w(p)$ より p は v_i から v_j への A-最短経路ではなく，矛盾が生じる．したがって，$w(p_1) = d(v_i, v_j, B)$ である．　□

A.2 補題 4 の証明

$p = (v_i, e_1, \ldots, e_l, v_j)$ を v_i から v_j への高さ最小の A-最短経路，すなわち，$h(p, A) = h(v_i, v_j, A)$ とする．$h(p, A) = h(v_i, v_j, A) = h$ とする．構文木の定義から，ある $A \to BC \in P$ とある v_k $(1 \leq k \leq l-1)$ に対して，p は $\max(h(p_1, B), h(p_2, C)) = h - 1$ を満たすB-経路 $p_1 = (v_i, e_1, \ldots, e_{l_1}, v_k)$ とC-経路 $p_2 = (v_k, e_{l_2}, \ldots, e_l, v_j)$ に分割できる．$\max(h(p_1, B), h(p_2, C)) = h - 1$ なので，以下の 2 条件のうち少なくとも 1 つが成り立つことを示せばよい．

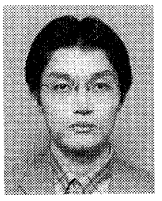
$$h(p_1, B) = h(v_i, v_k, B) = h - 1 \quad (4)$$

$$h(p_2, C) = h(v_k, v_j, C) = h - 1 \quad (5)$$

次の 3 つの場合に分けて考える．(i) $h(p_1, B) > h(p_2, C)$, (ii) $h(p_1, B) = h(p_2, C)$, (iii) $h(p_1, B) < h(p_2, C)$. 以下，(i) の場合を示す（他の場合も同様）．$\max(h(p_1, B), h(p_2, C)) = h - 1$ かつ $h(p_1, B) > h(p_2, C)$ なので，$h(p_1, B) = h - 1$ である．式 (4) が成り立つことを示すため，$h(v_i, v_k, B) = h - 1$ であることを示す．定義から $h(v_i, v_k, B) \leq h(p_1, B) = h - 1$ である．ここで，$h(v_i, v_k, B) < h - 1$ であると仮定する．(i) より $h(p_1, B) = h - 1 > h(p_2, C) \geq h(v_k, v_j, C)$ なので，$\max(h(v_i, v_k, B), h(v_k, v_j, C)) < h-1$ である．更に，定義から $h(v_i, v_j, A) \leq \max(h(v_i, v_k, B), h(v_k, v_j, C)) + 1$ である．よって $h(v_i, v_j, A) < h$ となり矛盾が生じる．したがって，$h(v_i, v_k, B) = h - 1$ である．　□

(平成 11 年 3 月 20 日受付)
(平成 11 年 6 月 27 日採録)

(担当編集委員 有川 正俊)



鈴木　伸崇（正会員）

1993 年大阪大学基礎工学部情報工学科卒業．1998 年奈良先端科学技術大学院大学情報科学研究科博士後期課程修了．同年，岡山県立大学情報工学部助手，現在に至る．博士（工学）．主にデータベース理論に興味をもつ．電子情報通信学会，ACM，IEEE-CS 各会員．

佐藤洋一郎（正会員）

昭57岡山大・工・電子卒．昭59同大大学院修士課程了．同年(株)東芝入社．昭62岡山大大学院博士課程入学．平1同中退後同大・工・情報助手．平7岡県大・情工助教授．工博．計算機ハードウェアに関する研究に従事．電子情報通信学会会員．

早瀬 道芳（正会員）

1968年京都大学工学部電子工学科卒業．1970年同大学院工学研究科修士課程修了．同年（株）日立製作所入社，中央研究所勤務．1993年より岡山県立大学情報工学部教授．博士（工学）．論理回路の自動設計，アルゴリズムの研究に従事．電子情報通信学会会員．