数理解析研究所講究録
731巻 1990年 286-297

リンク構造における検索処理効率と更新処理効率の関係について

九州大学工学部　木實新一 (Shinichi KONOMI)
九州大学大型計算機センター　古川哲也 (Tetsuya FURUKAWA)
九州大学工学部　上林彌彦 (Yahiko KAMBAYASHI)

1. まえがき

処理効率の良いデータベースを実現するためにリンク構造の導入を行っている例は多い. ネットワークデータベースはリンク構造に基づくものであり, オブジェクト指向データベースの参照関係や関係データベースの索引構造などもリンク構造であると考えられる. リンク構造を用いればリンクをたどるだけでデータの対応を求める検索を行うことができる. リンクが存在しなければデータの値を調べてデータの対応を求める必要があるので, これと比較すれば一般にリンク構造での検索は効率が良い.

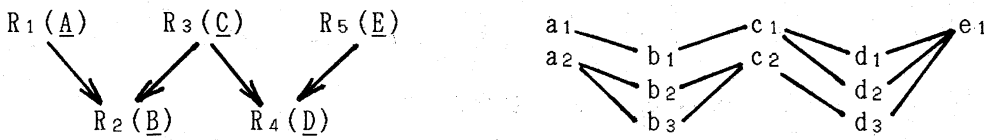
リンク構造では求めるデータの対応が, 直接あるいは少数のリンクを調べるだけで効率が良く求められる場合と, 多数のリンクを調べねばならなず効率が悪い場合がある. 効率良く求められるデータとそうでないものは, リンク構造の設計段階で決ってしまうため, 処理効率を考慮した設計を行う必要がある. 検索効率が良いスキーマを設計するためには, 一般に冗長な構造を付加するとよいが, 検索処理のみでなく更新処理まで考慮した場合, 加えた冗長な構造の更新処理コストがオーバヘッドとなる. 従って, 検索処理効率と更新処理効率とのトレードオフを考慮して, 処理効率をリンク構造に反映させることが可能なリンク構造の設計法が必要とされる.

検索処理と更新処理の効率のトレードオフを利用者の要求に基づいて決定するためには, 処理効率をなんらかの形で設計時に指定する必要がある. 文献[6,7]では, 検索処理効率を考慮したリンク構造の設計について, 文献[5]では

更新処理効率を考慮したリンク構造の設計について述べており，リンク構造と処理効率との適合性についての定性的な条件を用いて条件の選択などにより設計を行う方法を提案している．本稿では，処理効率とリンク構造の適合性に関するこれらの条件の関連を考察し，検索処理効率と更新処理効率のトレードオフを考慮したリンク構造の設計方法について述べる．

2．基本的事項

データベースにおけるリンク構造として代表的なものにネットワーク構造がある．ネットワーク構造（ネットワークスキーマ）は，バックマン線図と呼ばれる有向グラフ**B**(V,E)で表される．Vはレコードの集合であるレコード型に対応する節点の集合，Eはレコード間の一対多の対応を表す親子集合型に対応する有向枝の集合である．図1(a)にバックマン線図の例，(b)にその実現値の例を示す．ここで，属性集合Xからなるレコード型RをR(X)で表現している．また実現値では，属性のアルファベットに対応する小文字で表した属性値によりレコードを表現し，レコード間をリンクにあたる直線で結んでいる．図1(a)では属性A,B,…,Eからなるレコード型をそれぞれ$R_1, R_2, \cdots, R_5$，図1(b)では$R_1, R_2, \cdots, R_5$のレコードを$\{a_1, a_2\}, \{b_1, b_2, b_3\}, \cdots, \{e_1\}$で表している．



(a)　　(b)

図1　ネットワークスキーマとその実現値

下線の属性集合はレコード型のキーであり，その値を定めれば各レコード型でレコードが一意に定まる．値を定めれば**B**の全てのレコード型でレコードが一意に定まるような属性集合を**B**のキーと呼びK(**B**)で表す．一般にK(**B**)は**B**

中の子レコード型を持たないレコード型のキー集合の部分集合であり，図1(a)のキーはBDである．

ネットワーク構造に対する検索質問として，属性X_Sの値を指定し属性X_Pの対応する値を求める質問 $Q(X_S, X_P)$ を考える．この質問のクラスは，関係データベースにおける関係代数の選択(Selection)，射影(Projection)，結合(Join)により表現可能な，SPJ 質問と呼ばれる一般的な質問のクラスである．ここで，X_Sを選択属性，$X_Q = X_S \cup X_P$を質問属性と呼ぶ．検索質問Q を処理するためには，ネットワーク構造全体にわたる検索を行う必要は必ずしもなく，スキーマ中でX_Qを含むような部分のみを検索すれば十分である．このようなネットワーク構造の部分を質問Q に対する質問スキーマと呼び$\mathbf{B}_Q$で表す．$\mathbf{B}_Q$の性質は検索処理効率に影響を及ぼす．ネットワーク構造と検索処理との適合性について$\mathbf{B}_Q$の性質に基づいた次の条件がある[6,7]．

[条件１] 選択属性連結条件

$\mathbf{B}_Q$の全レコード型中で選択属性X_Sの値を指定するために調べる必要があるレコード型R(X)について $X \cap X_S \neq \phi$．Qの質問処理におけるX_Sの選択で，X_Sと共通属性を持たないレコード型を調べる必要がない．

[条件２] 質問属性連結条件

$\mathbf{B}_Q$中の全レコード型R(X)について $X \cap X_Q \neq \phi$．Qの質問処理で，X_Qと共通属性を持たないレコード型を検索する必要がない．

[条件３] 有向木条件

$\mathbf{B}_Q$ が有向木となる．効率が悪い親子集合型と逆方向の検索を行わなくてもよい．

[条件４] 非冗長解条件

$K(\mathbf{B}_Q) \subseteq X_Q$．質問の解が重複せず非冗長である．解の重複を取り除く必要がない．

スキーマに冗長なレコード型や親子集合型を加えたり，レコード型に冗長な属性を加えたりすることで，上記の条件を満たすような，検索処理効率の良いスキーマが得られる．以下のようなスキーマ変換の基本操作を組み合わせてこのようなスキーマ変換が可能になる[6,7]．図２は基本操作の例である．

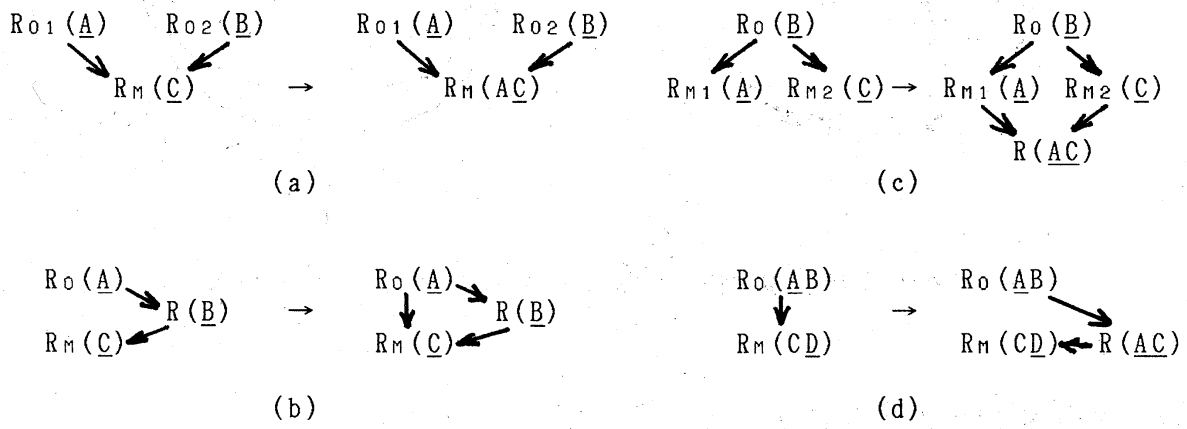


図２　スキーマ変換の基本操作例

(a) R_{O1}の属性AをR_Mに付加する．ABの対応はR_{O2},R_Mで求めることができる．

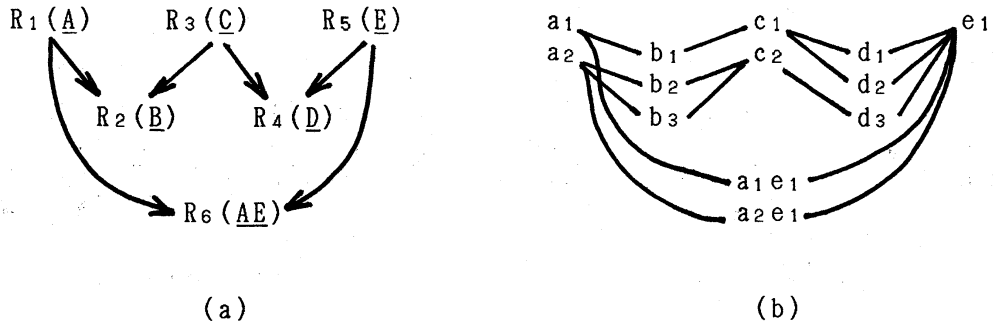
(b) R_OとR_M間にR を介するレコードの対応を表す親子集合型を付加する．ACの対応はR を検索しなくても求められ，検索するレコード型が削減される．

(c) R_{M1},R_O,R_{M2} の経路でのACの対応を表すレコード型R(AC)を付加する．ACの対応はRで直接得られる．

(d) R_OとR_Mとの間にレコード型R(AC)を加える．ABCの値はR,R_Oで求めることができ，この部分のキーはACとなるため解が重複しない．

[例１]　検索質問Q(A,E)を考える．図１のスキーマでは全てのレコード型を検索せねばA とE の対応は求めることができない．図３はQ の効率を考慮して図２(c) の基本操作により図１を変換したものである．冗長な構造として，図１の構造が表す属性Aと属性Eの値の対応を保持するレコード型R_6(AE)と，R_1及び

R_5からR_6に向かう親子集合型を加えることで，検索処理を効率化している．条件１～４を満たす図３では，レコード型$R_1, R_2, \cdots, R_5$ を検索する必要はなく，R_6の検索のみで属性AとEの対応を求めることができる．□



(a)　　(b)

図３　検索処理を考慮したスキーマとその実現値

３．更新処理を考慮したリンク構造

図２のような変換を用いて，あるリンク構造**B**に冗長な構造**B***を付加し検索効率の良いリンク構造を設計した場合，付加された冗長な部分においても更新処理が必要となってくる．このように，**B***の付加で冗長部の更新オーバヘッドが生じるが，この更新オーバヘッドには，単に冗長部のレコードやリンクを更新するコストだけでなく，どのレコードを更新するかなどを調べるための検索処理のコストも含まれる．

[例２] 図３のスキーマは検索処理を考慮し，冗長な構造**B***（レコード型R_6とそれに直接付随している親子集合型）を元の構造**B**（**B***以外の部分）に付加したスキーマである．ここでレコード型R_2のレコードb_2を削除する場合を考える．まずb_2を削除し，これとつながる**B***中のレコード型R_6のレコードを検索する．検索の結果a_2e_1が得られ，このレコードは**B**中でb_2を通らない経路でのレコードのつながりを持てば削除するがそうでなければ削除する必要はない．すなわちこのレコードは更新されるデータの候補である．レコードa_2e_1から再び検索を行うと，b_3を通る経路の存在が確かめられる．よって，a_2e_1を削除する必要

はない. □

このように，冗長な構造に更新を反映させるためには検索が必要である．更新には挿入，削除，修正が考えられるが，削除では例2のようにまず冗長な構造のデータのうち削除されたデータにリンクでつながるものを求め，次にそのデータが冗長な構造以外の部分に存在しないか調べた上で冗長な構造に更新を反映させる必要がある．挿入では新たに加えられたデータとリンクでつながる冗長な構造に挿入可能なデータをまず求め，次にそのデータがすでに冗長な構造中に存在しないか検索した上で更新を反映させる．修正は削除と挿入を組み合わせて考えることができる．

以上をまとめるとBでの更新を冗長な構造B*に反映させるために必要な処理は以下のようなになる．

(1) 被更新データ候補検索：B中の更新するデータがB*中のどのデータと対応しているかを調べるための検索処理

(2) 更新決定検索：(1)で求められたB*中のデータにB中の更新を反映させるか決定するための検索処理．

(3) 実更新：(2)での結果に応じて，(1)で求められたB*中のデータを実際に書き換える．（または書き換えない）

図4に冗長な構造が付加されたリンク構造における更新時の処理を分類して示す．

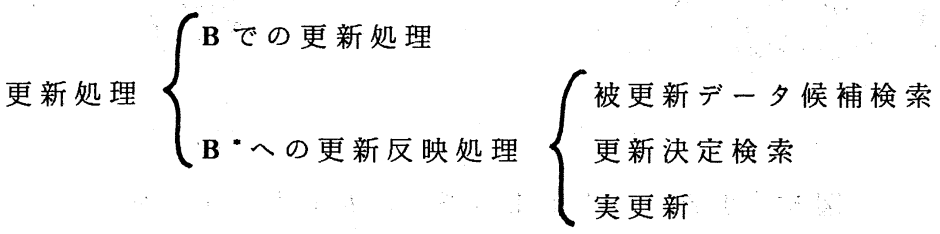


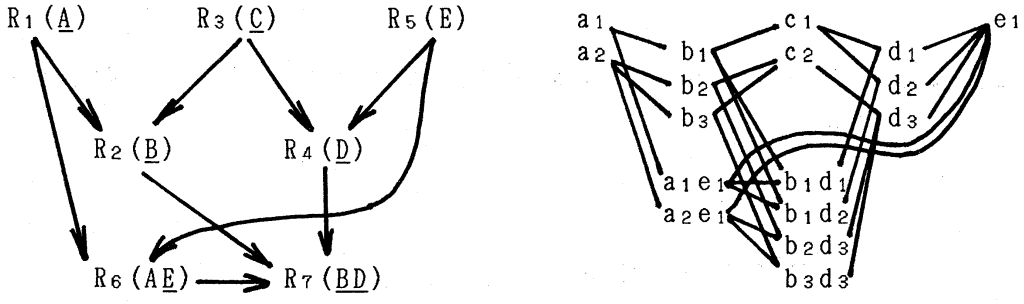
図4　冗長性のあるスキーマにおける更新時の処理

上に分類された処理のうちで，特に効率が問題となってくると考えられるのは更新決定検索であり，最悪の時間コストがスキーマのキーのデータ数n の自乗に比例する[5]．しかし，スキーマが次の条件を満たせば更新決定検索が不必要となり，更新時の処理は全体としてnに比例する時間で行える[5]．

[条件５] キー包含条件

冗長な構造$\mathbf{B}^*$には，属性集合が**B**の対応する部分の超キー（キーの超集合）となるレコード型が存在する．

[例３] 図３(a)のスキーマは，Q(A,E) の検索の効率は良いが，キー包含条件を満たさず更新処理オーバヘッドが大きい．これにR_7を付加した図５(a)のスキーマは，キー包含条件を満たし，$\mathbf{B}^*$中のキーを包含するレコード型R_7(BD)への更新反映は，更新するレコードとつながる R_7 のレコードを求め更新決定検索を行うことなく実更新すればよい．R_6の更新はR_7の更新を反映させることで局所的に容易に行うことができる．従って，図５(a) のスキーマでは更新処理オーバヘッドが小さく，Q(A,E)はR_6で図３と同様に容易に求めることができる．□



(a) (b)

図５ 更新処理と検索処理を考慮したスキーマ

4．更新及び検索に関する条件間の関連

検索処理とスキーマの適合性に関する１～４の条件と更新処理とスキーマの適合性に関する条件５との関連を考察する．特に関連が認められるのは，非冗長解条件（条件４）とキー包含条件（条件５）である．キー包含条件を満たす場合，被更新データ候補検索及び更新決定検索を検索質問として考えると，これらは非冗長解条件を満足する．

[定理１] スキーマがキー包含条件を満たせば，いかなる更新に対しても被更新データ候補検索及び更新決定検索の解は非冗長となる．□

[証明] 冗長な構造$\mathbf{B}^*$の属性集合をX^*とすれば，Aを更新する属性として，被更新データ候補検索は$Q(A, X^*)$である．求められたX^*の値に対し，更新決定検索は$Q(X^*, A)$であり，両者の検索の質問属性はともに$X_Q = A \cup X^*$である．条件５のキー包含条件が満たされれば$K(\mathbf{B}_Q) \subseteq X^*$，よって明らかに$K(\mathbf{B}_Q) \subseteq X^* \subseteq X_Q$であり，条件４の非冗長解条件が満たされる．すなわち，条件５は被更新データ候補検索と更新決定検索について条件４の非冗長解条件が満たされるための十分条件となる．（証明終わり）

更新決定検索については，更新時の処理におけるその目的が解が複数であるかどうかを調べることのみにあるため，非冗長解条件が満たされることが分かっていればこの検索を行う必要はなくなる．

非冗長解条件とキー包含条件の違いについて考察する．以下の２点はこの２つの条件の主な違いであると考えられる．

(1) 非冗長解条件は単一の検索の解が非冗長となるための条件であるが，キー包含条件が満たされる場合は複数の更新のための複数の検索の解が非冗長となる．

(2) キー包含条件ではキーを含むレコード型を作る必要があが，非冗長解条件ではそのようなことについて述べていない．

キー包含条件でキーを含むレコード型が存在せねばならぬのは，更新処理がレコード単位で行われるためである．冗長部のレコード及びリンクは一度に更新されるわけでなく，レコード単位で更新が行われるため，被更新データ候補検索Q(A,X*)及び更新決定検索Q(X*,A)は，実際にはレコード型R_iの属性をB_iとして被更新データ候補検索Q(A,B_i)と更新決定検索Q(B_i,A)として行われる．これらの検索はキー包含条件が満たされただけでは解が非冗長であることを保証されていない．しかし，キーを含むレコード型が存在すればキーを含むレコード型の更新時の検索は解が非冗長で，更新決定検索が不要である．その他のレコード型は，キーを含むレコード型を検索する事により更新時の検索を容易に行うことができる．従ってキーを含むレコード型が必要である．

5．検索処理効率と更新処理効率のトレードオフを考慮した設計

１～５の条件の選択に基づいてデータベースのスキーマを設計する方法は，必要な検索処理効率と更新処理効率を持つスキーマを設計する１つの手段である．しかし，要求に応じて検索処理効率と更新処理効率のトレードオフ問題をうまく解決するためには，条件を満たすかどうかという大まかな選択に基づく設計のみでは不十分で，より細かな選択に基づいた設計を行う必要がある．考えられる一つの方法は，キー包含条件を満たさない冗長な構造ができるだけ生じないようにして冗長な構造を加えていき検索効率のより良いスキーマを設計する方法で，これによってより細かくトレードオフに関する要求に応じて設計を行うことが可能である．

キー包含条件を満たすように注意して，属性連結に関する条件（条件１，２）を考慮し，冗長な構造を付加して検索処理効率の良いスキーマを設計する場合を例にとって考えると，トレードオフが異なる次のような４つの設計案が考えられる．

[設計１] 冗長な構造は付加しない．更新処理効率を非常に重視する場合にはこのようなスキーマを設計する必要がある．

[設計２] 効率化が必要な検索の経路上で，図２(b) の基本操作のような親子集合型の短絡路を設け得る部分があればすべて設ける．設計されるスキーマは，属性連結に関する条件を満たすとは限らなが，冗長な構造は親子集合型の短絡路のみであり更新が容易である．

[設計３] キーを包含するレコード型を作り，効率化が必要な検索の経路上で図２(b) の変換により親子集合型の短絡路を設け得る部分があればすべて設ける．設計されるスキーマは明らかにキー包含条件を満たすが，属性連結に関する条件を満たすとは限らない．全てのレコード型はキーを包含するレコード型との間に親子集合型を設けることが可能であるから，設計２の場合より検索処理を効率化することが可能である．

[設計４] キーを包含するレコード型を作り，効率化が必要な検索について属性連結にするために図２の基本変換の必要なものを用いる．冗長なレコード型が付加された場合はキーを包含するレコード型との間に親子集合型を設ける．設計されるスキーマは明らかにキー包含条件を満たし，属性連結に関する条件も満たされるスキーマである．従って検索処理の効率は４つの設計案のうちで最も良い．更新処理については，キー包含条件が満たされ更新決定検索は必要ないが，冗長な部分が４つの設計案のうちで最も多くなっており，それだけ更新処理コストは大きい．

[例４] 図６(a)のスキーマで，質問Q(A,G) を効率良く行う必要がある場合，設計１～４に基づいてそれぞれ設計を行えば(a)～(d)の様なスキーマが得られる．(c)(d)はキー包含条件を満たしており，(a)～(d)は全てスキーマのキーのデータ数に比例する手間で更新処理を行うことが可能である．

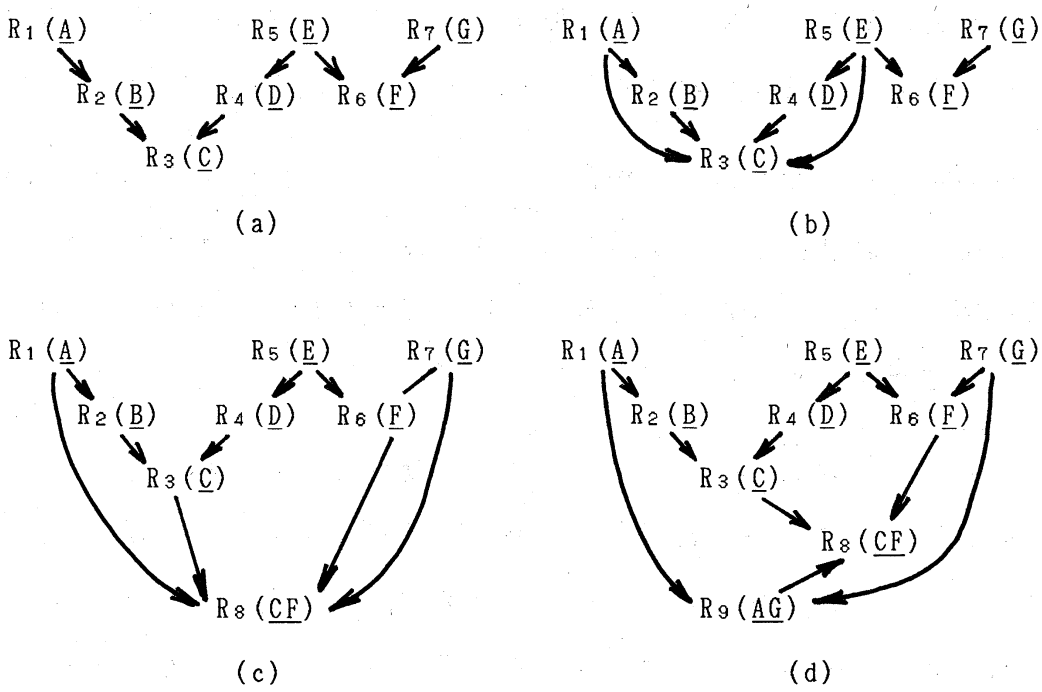


図6　トレードオフが異なる4つのスキーマ

更新処理の効率を非常に重視する場合は(a)のままでよい．但しQの検索では7個のレコード型を調べる必要がある．更新処理の効率をやや重視する場合は(b)のスキーマとする．(b)は図2(b)の親子集合型の短絡路を付加したもので，更新では冗長なリンクを局所的に管理すればよく，更新は容易であるが，Qの検索には5つのレコード型を調べる必要がある．Qの検索処理の効率をやや重視する場合は，(c)のスキーマとする．(c)はB_Qのキーからなるレコード型を加え，更に親子集合型の短絡路をつけたものである．新たに加えたレコード型はキー包含条件を満たし，更新決定検索が不要であるため，更新処理効率はそれほど悪くなく，Qの検索は3つのレコード型の検索でよい．Qの検索処理の効率をかなり重視する場合は(d)のスキーマとする．(d)では(c)に更に質問属性AGからなるレコード型を加えており，図5(a)と同様なスキーマである．Qは直接R_9で求められ，更新処理効率も，キー包含条件を満たしているのでそれほど悪くないが，更新を反映すべき冗長なレコード型が増加している．

6. むすび

検索処理と更新処理を効率良く行うことができるリンク構造を設計する際に用いることができる，スキーマと処理効率の適合性に関する条件[5,6,7]について考察し，非冗長解条件とキー包含条件間の関連と違いを明かにした．また，これらの条件のみを用いて表し得ない，更新処理と検索処理のトレードオフが異なるスキーマを示し，トレードオフを要求に応じてうまく決めることができるような設計方法を考察した．

本稿での考察などをもとにすれば，ある検索処理に対して検索処理効率と更新処理効率のトレードオフを，要求に応じてスキーマにかなり上手く反映できるようなリンク構造の設計法を開発できると思われるが，複数の検索と更新を同時に考慮する場合には更に考察が必要であると思われる．

参考文献

(1) Bipin, C.D., "Performance of a Composite Attribute and Join Index," IEEE Trans. on Software Eng., Vol.15, No.2, pp.142-152, Feb. 1989.

(2) Kuck, S.M. and Sagiv, Y., "Designing Globally Consistent Network Schemas," Proc. ACM SIGMOD Int. Conf. on Management of Data, pp.185-195, May 1983.

(3) Lien, Y.E., "On the Equivalence of Database Models," J. ACM, Vol.29, No.2, pp.333-362, April 1982.

(4) Valduriez, P., "Join Indices," ACM Trans. on Database Syst., Vol.12, No.2, pp.218-246, June 1987.

(5) 木實，古川，上林，“更新処理を考慮したリンク構造による物理スキーマ設計”，情報処理学会研究報告，89-DBS-74-1，平成元年11月．

(6) 古川，上林，“質問処理効率化のためのネットワークデータベースのスキーマ変換”，電子情報通信学会論文誌，Vol.J71-D，No.10，pp.2111-2119，昭和63年10月．

(7) 古川，上林，木實，“ネットワークデータベース設計支援システムの開発”，情報処理学会研究報告，88-DBS-67-5，昭和63年9月．