演繹データベースシステムにおける再帰的な問い合わせ処理の効率化に関する研究

１９９９年１月

鈴　木　晋

まえがき

演繹データベースシステムは，現在広く使われている関係データベースシステムに推論機構を付加したものである．関係データベースシステムの拡張として，また，人工知能分野における知識ベースシステムの核として期待されている．演繹データベースシステムの研究は1970年代後半から開始され，現在も，実用化に向けて様々な研究が進められている．

演繹データベースシステムは大量のデータを管理しており，利用者はこれらのデータに関し問い合わせを行う．システムは問い合わせが与えられると，問い合わせとデータから解集合を計算し，それを利用者に返す．問い合わせとデータの組を問題と言い，解集合を計算することを，問い合わせを処理する，あるいは，問題を解く，と言う．

演繹データベースシステムの長所の一つは，再帰的な問い合わせを処理できることである．関係データベースシステムはこれを処理することはできない．ところが，再帰的な問い合わせの処理には，素朴な方法により処理すると多大な計算時間を要する，という問題点がある．そのため，再帰的な問い合わせ処理の効率化が，演繹データベースシステムの重要な研究課題の一つとなっている．

再帰的な問い合わせ処理の効率化の研究は1980年代後半から活発に行われており，すでに，適用範囲は狭いが大変効率的な問い合わせ処理法から，前者ほど効率的ではないが適用範囲の広い問い合わせ処理法まで，様々な問い合わせ処理法が提案されている．しかし，問題の解決のためには，未だ十分ではないように思われる．

本研究は演繹データベースシステムにおける再帰的な問い合わせ処理の効率化に関する研究である．本研究では，再帰的な問い合わせ処理の効率化を更に進めるために，三つの新しい問い合わせ処理法（拡張HaNa法，直積法，緩和法と呼ぶ）を提案する．いずれの方法も，ホーン問題クラスのある部分クラスを対象としている．拡張HaNa法は適用範囲は狭いが大変効率的であり，一方，緩和法は拡張HaNa法ほどは効率的でないが適用範囲は広い．直積法は両者の中間的な適用範囲と効率をもつ．本研究では，また，直積法の内部処理に関連して基本的な組合せ問題（多次元直方体被覆問題と呼ぶ）を考え，この問題を効率的に解くためのアルゴリズムも提案する．このアルゴリズムを組み込んだ直積法は，適用範囲は狭くなるが，しかし，より効率的に問い合わせを処理することができる．

本論文は六つの章よりなる．

第1章は本論文の序論である．演繹データベースシステムの概要と再帰的な問い合わせ

処理の問題点を説明し，用語を定義した後，従来の代表的な問い合わせ処理法の概要を紹介する．また，従来の研究と比較しながら，本研究で提案する問い合わせ処理法およびアルゴリズムの概要を説明する．

第2章から第5章までは本論文の本論である．ここでは，本研究（すなわち，拡張HaNa法，直積法，緩和法，多次元直方体被覆問題を解くアルゴリズム）を詳しく説明する．

第2章では拡張HaNa法（文献[SIK92, SIK93]）を説明する．同世代問題はある大変限定された形をした問題である．同世代問題の中で，再帰述語の引数の数mが2の場合の問題は有名な問題であり，多くの研究が行われている．$m = 2$の場合の同世代問題を効率的に解くことを目的として，計数法やHaNa法等が提案され，そして，これらの方法および広い範囲の問題を対象とするマジック集合法の最悪時間量が解析されている．$m = 2$の場合，最悪時間量において，HaNa法およびマジック集合法が最も効率的である．一方，$m \geq 3$の場合の同世代問題についてはあまり研究が行われていない．$m \geq 3$の場合，従来の方法の中で，最悪時間量において，マジック集合法が最も効率的であると思われる．本研究では，$m \geq 3$の場合の同世代問題が効率的に解けるようにHaNa法を変形して，拡張HaNa法を提案する．拡張HaNa法の最悪時間量を解析し，$m \geq 3$の場合，通常よく生じるデータに対して拡張HaNa法がマジック集合法より効率的であることを示す．

第3章では直積法（文献[SIK90]の一般化）を説明する．上の同世代問題のクラスは狭い問題クラスであるが，一方，広い問題クラスにはdatalog問題クラスがある．datalog問題クラスは同世代問題クラスを包含している（“同世代問題クラス” ⊂ “datalog問題クラス” ⊂ “ホーン問題クラス”）．第3章では，同世代問題クラスを包含し，そして，一部の非線形問題も含む，datalog問題クラスの新しい部分クラス（直積問題クラスと呼ぶ）を定義する．直積法は直積問題クラスを解くための問い合わせ処理法である．直積問題クラスは比較的広いクラスであるので，従来の方法の中で直積問題クラスに適用できるのは広い適用範囲をもつ方法だけである．従来の方法の中では，マジック集合法が最も効率的であると思われる．直積問題の例として，同世代問題の再帰ルールを複数にした問題，および，同世代問題の再帰ルールを非線形にした問題を考え，これらの問題を使って，直積法とマジック集合法の効率を比較する．最悪時間量において，直積法はマジック集合法より非効率である．しかし，計算機実験によって，データが密な場合，直積法がマジック集合法より効率的であることを示す．

第4章では多次元直方体被覆問題を解くアルゴリズム（文献[SI97]）を説明する．直積問題クラスの部分クラスである連続直積問題クラスを定義する．多次元直方体被覆問題は，連続直積問題に直積法を適用した際に，直積法の内部処理に現れる基本的な組合せ問題である．多次元直方体被覆問題は，組合せ問題の分野において有名な充足可能性問題の一般化でもある．新しいアルゴリズムを提案すること，および，充足可能性問題のための従来のアルゴリズムを多次元直方体被覆問題に素直に拡張することにより，多次元直方体被覆問題を効率的に解くための複数のアルゴリズムを与える．そして，計算機実験により，これらのアルゴリズムが多次元直方体被覆問題を効率的に解くことを示す．これらのアルゴリズムを直積法に組み込むと，直積法の適用範囲は連続直積問題クラスに狭まるが，しかし，直積法の効率は更に高まる．

第5章では緩和法（文献[SIK91, SIK94]）を説明する．datalog問題クラスを包含する広い範囲の問題を効率的に解くための方法には，すでに名前をあげたマジック集合法やその他多くの方法がある．これらの方法では，制約を求め，その制約を使って問い合わせに無関係なデータの生成をできるだけ防ぐことにより，問い合わせを効率的に処理する．緩和法は，これらの方法と同様，制約を使って問い合わせを効率的に処理するが，しかし，制約の求め方がこれらの方法より柔軟である．マジック集合法の一つであるマジックテンプレート法は，広い範囲の問題を効率的に解くための従来の方法の中で最も適用範囲が広く，datalog問題クラスと一部のホーン問題を解くことができ，かつ，最も効率的である（ただし，アレクサンダーテンプレート法とHCT/R法もマジックテンプレート法と同じ能力を持つ）．マジックテンプレート法と緩和法を比較し，マジックテンプレート法が解くことができる任意の問題を緩和法が少なくとも同じ効率で解けること，マジックテンプレート法が解けるある問題を緩和法がより効率的に解けること，マジックテンプレート法が解けないある問題を緩和法が効率的に解けること，を示す．なお，緩和法の適用範囲はdatalog問題クラスと一部のホーン問題であるが，ホーン問題部分がマジックテンプレート法より広い．

上に述べた各問題クラスの包含関係を整理すると，(“同世代問題クラス”，“連続直積問題クラス”) ⊂ “直積問題クラス” ⊂ “datalog問題クラス” ⊂ “緩和法の適用範囲” ⊂ “ホーン問題クラス”，になる．

第6章は本論文の結論である．本研究のまとめを述べる．

謝辞

本研究を行い，まとめるにあたり，長年に渡り，懇切なる御指導と多大な御尽力をして頂きました京都大学大学院工学研究科茨木俊秀教授に心より感謝の意を表します．茨木先生に本研究を御指導して頂いて以来すでに11年の歳月が過ぎました．また，学生時代に卒業研究を御指導して頂いてから数えると20年の歳月が過ぎました．この間ずっと，愚鈍な私を忍耐強く我慢強く御指導して頂き，多大な御迷惑をおかけしましたにも関わらず，終始暖かく励まして下さいました．心より深謝致します．

京都大学大学院工学研究科上林彌彦教授，京都大学大学院工学研究科加藤直樹教授には茨木先生と共に本研究を審査して頂き，貴重な御助言を数多く頂きました．心より感謝致します．また，京都大学大学院工学研究科岩井敏洋教授，京都大学大学院工学研究科福嶋雅夫教授には私の学力を審査して頂き，本研究につきましても貴重な御助言を頂きました．心より感謝致します．

京都大学大学院工学研究科長谷川利治教授（現京都大学名誉教授，南山大学教授）には，学生時代以来，茨木先生と共にずっと御指導をして頂きました．公私ともども多くの御援助と励ましの御言葉を頂きました．長谷川先生に心より深謝致します．

私の職場である愛知工業大学情報通信工学科の教職員の方々には，愚鈍な私が本研究を少しでも早くまとめることができますように，ここ数年，雑用を減らす等の御配慮をして頂きました．御迷惑をおかけしましたことを心より御詫び致しますと共に，御配慮に対し深く感謝致します．また，情報通信工学科以外の先生方にも御心配をして頂きました．深く感謝致します．

最後に，安らぎと喜びを与えてくれた妻宏子と子供達，心配をしてくれた母昌代に感謝致します．

目 次

第１章

序論

本論文は，演繹データベースシステムにおける再帰的な問い合わせ処理の効率化に関する論文である．本論文では，再帰的な問い合わせ処理を効率化するために，三つの新しい問い合わせ処理法（拡張HaNa法，直積法，緩和法と呼ぶ）を提案し，また，直積法の内部処理に関連して基本的な組合せ問題（多次元直方体被覆問題と呼ぶ）を考え，この問題を効率的に解くためのアルゴリズムも提案する．

本章は本論文の序論である．本章では，演繹データベースシステムの概要を説明した（第1.1節）後，再帰的な問い合わせ処理の効率化がなぜ重要であるのかを説明する（第1.2節）．そして，準備として用語を定義した（第1.3節）後，再帰的な問い合わせを効率的に処理するための従来の代表的な幾つかの方法について，その概要を紹介する（第1.4節）．最後に，従来の研究との位置づけを示しながら，本研究で提案する問い合わせ処理法およびアルゴリズムの概要を説明する（第1.5節）．

1.1　演繹データベースシステムの概要

演繹データベースシステムは，現在広く使われている関係データベースシステムに推論機構を付加したものである．関係データベースシステムの拡張として，また，人工知能分野における知識ベースシステムの核として期待されている．演繹データベースシステムの研究は1970年代後半から開始され，現在も，実用化に向けて様々な研究が進められている[GM78, AU79, GMN 84, Ull85, Min88, Ull88, Ull89, Kane90, Kiyo90, Koba90, Ull90, UZ90, Zani90, Ull91]．

演繹データベースシステムは大量のデータを管理しており，利用者はこれらのデータに

関し問い合わせを行うことができる．システムは問い合わせが与えられると，問い合わせとデータより解集合を求め，それを利用者に返す．

以下の例に示すように，演繹データベースシステムでは，問い合わせはルールと，解として必要なデータの形を表す式を使って記述する．また，問い合わせに対する解集合は，システムが管理しているデータに問い合わせ中のルールを適用したときに生成されるすべてのデータの中で，問い合わせ中の式と形が一致するデータの集合である．

例 1.1 親子関係を表すデータと，そこに現れる人を表すデータの集合

$$\begin{aligned} D1 \quad = \quad & \{Par(b,a), Par(c,b), Par(d,c), \ldots\} \\ & \cup \{Person(a), Person(b), Person(c), Person(d), \ldots\} \end{aligned}$$

が演繹データベースシステムに蓄えられている．ここで，データ $Par(b,a)$ は「b が a の親である」ことを，データ $Person(a)$ は「a が親子関係に現れる人である」ことを表し，他のデータも同様とする．ある共通先祖より同じ世代数だけ下がった子孫同士は“同世代である”と言う．利用者はこのデータ集合において，ある特定の人 b と同世代であるすべての人を知りたい．

このような場合，利用者は，演繹データベースシステムに対し，次のように問い合わせを行うことができる．まず，「X と Y が同世代である」ことを意味する $sg(X,Y)$ を人 $Person$ と親子関係 Par を使って記述するために，例えば次のルール集合 $P1$: (1.1), (1.2), を作る．

$$P1: \quad sg(X,X) \leftarrow Person(X). \tag{1.1}$$

$$sg(X,Y) \leftarrow Par(X_p,X), sg(X_p,Y_p), Par(Y_p,Y). \tag{1.2}$$

ここで，X, Y, X_p, Y_p は変数であり，ルール (1.1) は「X は自分自身 X と同世代である」ことを，ルール (1.2) は「X_p と Y_p が同世代であり，かつ，X_p が X の親であり，かつ，Y_p が Y の親であれば，X と Y は同世代である」ことを表す．次に，データ集合 $D1$ のデータにルール集合 $P1$ のルールを適用したときに生成されるすべての sg データ（すなわち，同世代関係データ）の中で，ある特定の人 b と同世代である人だけを知りたいことを示すために，式 $sg(b,Y)?$ を作る．そして，ルール集合 $P1$ と式 $sg(b,Y)?$ をシステムに与える．

演繹データベースシステムには，すなわち，データ集合 $D1$，ルール集合 $P1$，式 $sg(b,Y)?$ が与えられる．データ集合 $D1$ の例として，次のデータ集合を考える（親子関係 Par を図

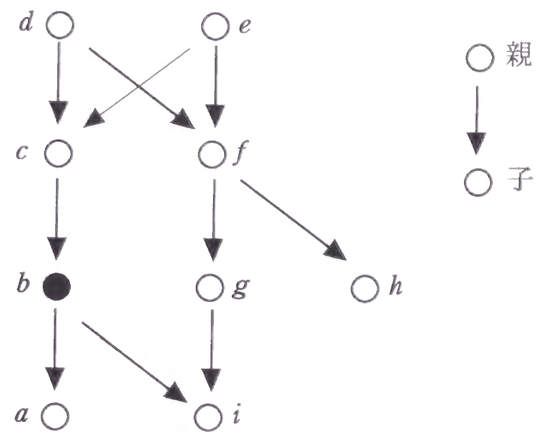


図 1.1: 親子関係を表すグラフ $G1$

示すると図 1.1 になる）．

$$\begin{aligned} D1 \quad = \quad & \{Par(b,a), Par(c,b), Par(d,c), Par(e,c), \\ & Par(d,f), Par(e,f), Par(f,g), Par(f,h), Par(b,i), Par(g,i)\} \\ & \cup \{Person(a), Person(b), Person(c), Person(d), Person(e), \\ & Person(f), Person(g), Person(h), Person(i)\} \end{aligned} \tag{1.3}$$

このとき，問い合わせに対する解集合は次のようなデータの集合である．例えば，$D1$ の $Person(d)$ にルール (1.1) を適用すると $sg(d,d)$ が生成される．その $sg(d,d)$ と $Par(d,c)$, $Par(d,f)$ にルール (1.2) を適用すると $sg(c,f)$ と $sg(f,c)$ が生成される．更に，例えば，その $sg(c,f)$ と $Par(c,b)$，$Par(f,g)$ にルール (1.2) を適用すると $sg(b,g)$ が生成される．もし，$D1$ のデータに対する $P1$ のルールのこのような適用を新しい sg データが生成できなくなるまで可能な限り繰り返すならば，同世代関係を表すデータ集合

$$\begin{aligned} IMP1 \quad = \quad & \{sg(d,d), sg(e,e), \\ & sg(c,c), sg(c,f), sg(f,c), sg(f,f), \\ & sg(b,b), sg(b,g), sg(b,h), sg(g,b), sg(g,g), sg(g,h), \\ & sg(h,b), sg(h,g), sg(h,h), \end{aligned}$$

$$sg(a,a),\ sg(a,i),\ sg(i,a),\ sg(i,i)\}$$

が生成される．問い合わせに対する解集合は，$IMP1$のデータの中で式$sg(b,Y)?$と形が一致するデータ，すなわち，第1引数の値がbであるデータの集合

$$ANS1 \quad = \quad \{sg(b,b),\ sg(b,g),\ sg(b,h)\}$$

である．演繹データベースシステムは様々な方法でこの解集合$ANS1$を計算し，それを利用者に返す．□

例1.1において，演繹データベースシステムが管理しているデータの集合$D1$を**ファクト集合**（fact set），ルール集合とファクト集合の組$S1 = (P1, D1)$を**演繹データベース**（deductive database, ddbと略す），集合$IMP1$を演繹データベース$S1$の**意味**（semantics）と言う（意味にファクト集合を含める場合もあるが，本研究では，説明の簡単のため，意味にファクト集合を含めない）．意味$IMP1$は演繹データベース$S1$が定める仮想的なデータ集合である．利用者が知りたいものを記述するためのルール集合と式の組，すなわち，$(sg(b,Y)?, P1)$を**問い合わせ**（query）と言う．また，利用者が知りたいものを記述する式，すなわち，$sg(b,Y)?$を本論文では**問い合わせアトム**（query atom）と言う．問い合わせとファクト集合の組，すなわち，問い合わせアトムと演繹データベースの組$(sg(b,Y)?, S1)$を**問題**（problem）と言う．集合$ANS1$を**解集合**（answer set）と言う．解集合$ANS1$は，意味$IMP1$に含まれるデータの中で，問い合わせアトム$sg(b,Y)?$と形が一致するデータの集合である．解集合を求めることを**問い合わせを処理する**（evaluate a query），あるいは，**問題を解く**（solve a problem）と言う．

なお，例1.1の問題$(sg(b,Y)?, S1)$は有名な問題であり，**同世代問題**（same generation problem）と呼ばれている．

1.2 再帰的な問い合わせの処理とその問題点

例1.1では，ルール(1.2)の"←"の左側および右側に述語sgが現れ，述語sgが述語sgを使って定義されている．そのため，述語sgおよびルール(1.2)は**再帰的である**（recursive）と言う（詳しくは第1.3節を参照）．また，再帰的な述語（ルール）を含んでいるので，問い合わせも**再帰的である**と言う．

演繹データベースシステムと関係データベースシステムを比較した場合，演繹データベースシステムの長所の一つは，例1.1に示したように，再帰的な問い合わせを扱えることである．関係データベースシステムは再帰的な問い合わせを扱うことはできない．関係データベースシステムでは，関係代数や関係論理等の問い合わせ言語を使って問い合わせを記述するが，これらの言語を使って再帰的な問い合わせを記述しようとすると，記述式の長さが無限になるからである．

ところが，再帰的な問い合わせの処理には，素朴な方法により処理すると多大な計算時間を要する，という問題点がある．ここで，素朴な方法とは，意味IMPを実際に生成し，その中から解集合ANSを選択する方法であり，**セミナイーブ法**（semi-naive method）[FU76, PS77, BR86]（第1.4.1節を参照）と呼ばれている．多大な計算時間を要するのは次の理由による．再帰的な問い合わせでは，通常，IMPの大きさは爆発的に大きくなる．例えば，例1.1において，データ集合$D1$の中の異なる人の数をnとすると，通常，$D1$の大きさが$O(n)$程度であるのに対し，$IMP1$（同世代関係のデータ集合）の大きさは$O(n^2)$になる．nが大きい場合，$O(n^2)$は爆発的に大きい．また，IMPを実際に生成するには，少なくともIMPの大きさの計算時間を必要とする（第1.4.1節では計算時間をより正確に評価する）．そのため，再帰的な問い合わせをセミナイーブ法を使って処理すると，IMPの生成に多大な計算時間を要し，その結果，ANSを得るのに多大な計算時間を要する．

このように，演繹データベースシステムでは，再帰的な問い合わせ処理の効率化が重要な研究課題の一つとなっている．

1.3 演繹データベースの構文

従来の問い合わせ処理法および本研究で提案する問い合わせ処理法の概要を説明するための準備として，演繹データベースの構文に関し，用語を幾つか定義する．

項（term）は，(a) 定数は項である，(b) 変数は項である，(c) $t_1, t_2, \ldots, t_k$が項でfがk引数関数記号ならば式$f(t_1, t_2, \ldots, t_k)$は項である，(d) その他のいかなる式も項ではない，により帰納的に定義される式である．例えば，a, bを定数，X, Yを変数，fを1引数関数記号，gを2引数関数記号とするとき，$a, X, f(a), f(f(a)), f(X), f(f(X)), g(a,b), g(g(X,Y), f(f(X)))$等が項である．変数を含まない項を**基礎項**（ground term）と言う．例えば，上記の項の例のうち，$a, f(a), f(f(a)), g(a,b)$が基礎項である．

m個の基礎項の組の集合$\{(t_1, t_2, \ldots, t_m) \mid t_i, i = 1, \ldots, m,$は基礎項$\}$から集合$\{$真,偽$\}$への写像を$m$引数**述語**（predicate）と言う．述語には通常述語と算術述語があり，通常述語には，更に，IDB述語とEDB述語がある．**通常述語**（ordinary predicate）はルール集合やファクト集合により定義される述語である．ルールにおいて，記号“←”の左側を**頭部**（head），右側を**本体**（body）と言う．一般性を失うことなく，ルールの頭部とファクト集合の両方に現れる述語はないと仮定する．**IDB述語**（intensional database predicate）はルール集合により定義される（すなわち，ルールの頭部に現れる）述語であり，**EDB述語**（extensional database predicate）はファクト集合により定義される（すなわち，ファクト集合に現れる）述語である．例えば，例1.1のsgはIDB述語，$Person$とParはEDB述語である．**算術述語**（arithmetic predicate）は，$X > Y$や$X = Y + 1$のような算術比較式により定義される述語である（算術述語はルールの頭部にもファクト集合にも現れない）．

$t_1, t_2, \ldots, t_m$が項，pがm引数述語のとき，式$p(t_1, t_2, \ldots, t_m)$を**アトム**（atom）と言う．pを通常述語，$>$を算術述語，X, Yを変数，a, bを定数，fを関数記号とするとき，$p(X, Y)$, $p(a, b), p(f(X), f(Y)), > (X, Y)$等はアトムである．また，例1.1の$sg(X, Y)$や$Person(X)$, $Par(X, X_p)$，問い合わせアトム$sg(b, Y)$?もアトムである（問い合わせアトムについては，問い合わせアトムであること陽に示すために，“?”を付けている）．通常（IDB，EDB，算術）述語のアトムを通常（IDB，EDB，算術）アトムと言う．算術アトム$> (X, Y)$は，通常の数学記法にしたがって，$X > Y$と記すこともある．アトムのうち，変数を含まないアトムを**基礎アトム**（ground atom）と言い，gaと略記する．例えば，先のアトム$p(X, Y), p(a, b), p(f(X), f(Y)), X > Y$の中で，$p(a, b)$はgaであるが，それ以外はgaではない．また，例1.1の$Person(a)$や$Par(b, a)$，$sg(b, g)$はgaである．ファクト集合に含まれるgaを**ファクト**（fact）と言う．例1.1の$Person(a)$，$Par(b, a)$はファクトであるが，$sg(b, g)$はファクトではない．

アトム$p(t_1, t_2, \ldots, t_m)$の中の項$t_i, i = 1, \ldots, m,$をそのアトムの**第i引数**と言う．

以後，特に断らない限り，定数は$a, b, \ldots, a_1, \ldots$のように小文字で始まる文字列により，変数は$X, Y, \ldots, X_1, \ldots$のように大文字で始まる文字列により，EDB述語は$A, B, \ldots, Par, \ldots$のように大文字で始まる文字列により，IDB述語は$p, q, \ldots, sg, \ldots$のように小文字で始まる文字列により表す．

$\boldsymbol{t}, \boldsymbol{t}_1, \ldots, \boldsymbol{t}_n$を引数ベクトル，$p(\boldsymbol{t})$を通常アトム，$q_1(\boldsymbol{t}_1), \ldots, q_n(\boldsymbol{t}_n)$をアトムとするとき，式

$$p(\boldsymbol{t}) \leftarrow q_1(\boldsymbol{t}_1), \ldots, q_n(\boldsymbol{t}_n).$$

を**ホーンルール**（Horn rule）と言う．ホーンルールは「$q_1(\boldsymbol{t}_1)$かつ…かつ$q_n(\boldsymbol{t}_n)$ならば$p(\boldsymbol{t})$である」ことを表す．ホーンルールの中で，算術アトムも関数記号も含まないルールを**datalogルール**と言う．

例として，ルール

$$p(X, Y) \leftarrow q(X, Z), p(Z, Y). \tag{1.4}$$

$$p(a, b) \leftarrow . \tag{1.5}$$

$$p(X, Y) \leftarrow q(a, Z), p(Z, Y), X > Y. \tag{1.6}$$

$$p(f(X), f(Y)) \leftarrow p(X, Y). \tag{1.7}$$

$$p(X, Z) \leftarrow A(X, Y), p(Y, Z), \neg q(X, Z). \tag{1.8}$$

$$p(X, Z) \vee s(X, Z) \leftarrow A(X, Y), p(Y, Z). \tag{1.9}$$

を考える．ここで，fは関数記号，$\neg$は否定，$\vee$は論理和である．これらのルールのうち，(1.4)から(1.7)はホーンルールであるが，(1.8)は，否定（$\neg q(X, Z)$）を含むために，ホーンルールではなく，(1.9)は，頭部が二つのアトムの論理和（$p(X, Z) \vee s(X, Z)$）であるために，ホーンルールではない．また，ホーンルール(1.4)から(1.7)のうち，(1.4)と(1.5)はdatalogルールであるが，(1.6)は，算術アトム（$X > Y$）を含むために，datalogルールではなく，(1.7)は，関数記号（f）を含むために，datalogルールではない．例1.1のルール(1.1)と(1.2)はともにdatalogルールである．

演繹データベース$S = (P, D)$のルール集合Pのすべてのルールがホーンルール（datalogルール）であるとき，**ホーン演繹データベース（datalog演繹データベース）**と言う．また，問題のddbがホーン演繹データベース（datalog演繹データベース）であるとき，**ホーン問題（datalog問題）**と言う．

演繹データベースはホーン演繹データベースであると仮定する．ルール集合Pから次のようにグラフGを作る．Gの節点はPの述語である．Gの枝は，述語qを頭部に，述語pを本体にもつルールがあるとき，枝(p, q)がある．このように定義されたグラフGに閉路がある場合，閉路上にある述語は**再帰的である**（recursive）と言い，同じ閉路上にある二つの述語同士は**相互再帰的である**（mutually recursive）と言う．ここで，再帰的，相互再帰的な述語はすべてIDB述語である．ルールの頭部の述語とそのルールの本体のある述語が相互再帰的であるとき，そのルールは**再帰的である**と言う．例えば，例1.1を考えると，第1.2節に述べたように，述語sgおよびルール(1.2)は再帰的である．また，IDB述語p, q, sと

EDB述語A, Bよりなる次のルール集合

$$P: \quad p(X,Y) \leftarrow A(X,Y). \tag{1.10}$$

$$q(X,Z) \leftarrow B(X,Y), p(Y,Z). \tag{1.11}$$

$$p(X,Z) \leftarrow q(X,Y), q(Y,Z). \tag{1.12}$$

$$s(X,Z) \leftarrow p(X,Y), q(Y,Z). \tag{1.13}$$

を考えると，グラフGにpからqを経てpに戻る閉路があるので，述語pとqは再帰的であり，そして，ルール(1.11)と(1.12)は再帰的である．

頭部の述語と相互再帰的な述語が本体に高々一カ所しか現れないホーンルールは**線形である**（linear）と言い，それ以外のホーンルールは**非線形である**（nonlinear）と言う．よく起きる例として，ホーンルールの本体にIDB述語が高々1箇所しか現れないならば，そのルールは線形になる．例えば，例1.1のルール(1.1)と(1.2)はともに線形である．また，上記のルール(1.10)から(1.13)において，ルール(1.10)，(1.11)，(1.13)は線形であるが，ルール(1.12)は非線形である．

1.4 従来の問い合わせ処理法の概要

再帰的な問い合わせ処理の効率化を目的とした研究は1980年代後半から活発に行われている．Bancilhon, Beeri, Han, Henschen, Lozinskii, Naqvi, Naughton, Ramakrishnan, Rohmer, Sacca, Shapiro, Ullman, Vieille, Wong, Zaniolo らにより，また，日本では第五世代コンピュータプロジェクト（ICOT）が中心となり，世木，西尾，宮崎，横田らにより多くの成果が報告されている[MS81, Ban85, Loz85, Gel86, Vie86, BKBR87, IW87, Han88, Nau88]．これらを要約した文献には[BR86, NK88, Ull88, Ull89, Kat90, MS90, Ull90, Mori95, RU95]等がある．

再帰的な問い合わせ処理の効率化の研究では，datalog問題クラス，または，その部分クラスを対象とする研究が多い．

datalog問題クラスを研究対象とするのは次の理由による．datalog問題クラスの問題では，セミナイーブ法を使えば，ddbの意味IMPを有限時間で計算することができ，故に，問題の解集合ANSを有限時間で計算することができる．一方，ホーン問題クラスの問題では，第1.3節のルール(1.6)や(1.7)のように，ルールが算術アトムや関数記号を含む場合，

ddbの意味IMPが無限集合になる場合があり，故に，解集合ANSも無限集合になる場合がある．また，ホーン問題クラスより広い問題クラスの問題では，第1.3節のルール(1.8)や(1.9)のように，ルールが否定や論理和を含む場合，ddbの妥当な意味IMPを定義すること自体が困難な場合があり，故に，解集合ANSを定義することが困難な場合がある．以上が理由である．

また，datalog問題クラスの部分クラスを研究対象とするのは次の理由による．datalog問題のクラスの中にも様々な部分クラスがあり，部分クラスが異なると，問い合わせ処理の効率化のために利用できる手法が異なる．部分クラスが狭くても，その部分クラスが応用上よく現れるものであり，かつ，その部分クラスを解くための，datalog問題クラスを解くアルゴリズムよりももっと効率的なアルゴリズムが得られるならば，その部分クラスは重要である．これが理由である．

なお，datalog問題クラスを解くことを目的に提案された問い合わせ処理法は，たとえ問題がdatalog問題ではないホーン問題であっても，ある問題に対しては，その解集合を計算できる場合がある．このため，本論文では，そのような問い合わせ処理法の適用範囲は，datalog問題クラスと一部のホーン問題である，として説明する．

本節の以降では，datalog問題クラスと一部のホーン問題を対象とする従来の代表的な問い合わせ処理法，および，datalog問題クラスの部分クラスである同世代問題クラスを対象とする従来の代表的な問い合わせ処理法について，その概要を紹介する．

1.4.1 基本的な問い合わせ処理法（セミナイーブ法）の概要

セミナイーブ法（semi-naive method）[FU76, PS77, BR86]は最も基本的な問い合わせ処理法である．第1.2節で述べたように，意味IMPを生成し，その中から解を選択する．セミナイーブ法の適用範囲は意味IMPが有限集合である任意のホーン問題，すなわち，datalog問題クラスと一部のホーン問題である．セミナイーブ法は単独で用いられることもあるが，多くは，第1.4.2節，第1.4.3節で示すように，効率化を目的とした他の問い合わせ処理法の中で用いられる．

セミナイーブ法は，ファクトおよび/または生成されたgaよりなるga組で，ルールの本体に代入可能なga組をルールの本体に代入することにより，頭部についてのgaを生成する．例えば，第1.1節の例1.1の問題において，ファクト$Par(d,c)$と$Par(d,f)$があり，ga $sg(d,d)$が生成されている場合，それらよりなるga組，例えば，$(Par(d,c), sg(d,d), Par(d,f))$は

$\Delta I = D$;
$I = D$;
while $(\Delta I \neq \phi)\{$
　$\Delta I = \cup_{r \in P} Eval(r, \Delta I, I) - I$;
　$I = I \cup \Delta I$;
$\}$;
$IMP = I - D$;
$ANS = \{q(\boldsymbol{a}, \boldsymbol{x}) \mid q(\boldsymbol{a}, \boldsymbol{x}) \in IMP\}$;

図1.2: セミナイーブ法

ルール(1.2)の本体$Par(X_p, X) \wedge sg(X_p, Y_p) \wedge Par(Y_p, Y)$に代入可能であるので，それをルール(1.2)の本体に代入して，頭部についてのga $sg(c, f)$を生成する．セミナイーブ法は，このようなga組の代入によるgaの生成を，新しいga組が作れなくなるまで繰り返す．また，同じルールに同じga組を重複して代入しても同じgaしか生成されず，無駄であるので，ga組の代入に当たっては，このような重複が起こらないように工夫している．セミナイーブ法は，このようにしてga組をルールの本体に代入することでgaを生成し（ここで，生成されたgaの集合が意味IMPになる），生成されたgaの中から解を選択する．

問題を$(q(\boldsymbol{a}, \boldsymbol{X})?, (P, D))$として，セミナイーブ法を図1.2に示す．

図において，$Eval(r, \Delta I, I)$は，ga集合Iのgaよりなり，少なくとも一カ所にはΔI（$\subseteq I$）のgaを含むga組で，かつ，ルールrの本体に代入可能であるga組を重複したり洩れたりすることがないように計算しながら，そのga組をルールrの本体に代入して頭部のgaを生成していき，そして，そのように生成されたすべてのgaの集合を返すサブルーチンである．

whileループの中の$\{\Delta I = \cup_{r \in P} Eval(r, \Delta I, I) - I;\ I = I \cup \Delta I;\}$部分をwhileブロックと呼ぶ．$k$回目のwhileブロックの実行終了時点を考えると，Iにはファクトおよびそれまでに（すなわち，k回目までのwhileブロックの実行において）生成されたgaの集合が入っており，ΔIにk回目のwhlieブロックの実行において新しく生成されたgaの集合が入っている．$\Delta I \neq \phi$の場合，$(k+1)$回目のwhlieブロックが実行され，サブルーチン$Eval(r, \Delta I, I)$において，Iのga（すなわち，ファクトおよび/またはそれまでに生成されたga）よりなるga組で，少なくとも1箇所にΔIのga（すなわち，k回目のwhlieブロックの実行において新しく生成されたga）を含む各ga組が計算され，そして，それがルールrの本体に代入されて，頭部についてのgaが生成される．ここで，ga組を，少なくとも1箇所にΔIのgaを含むga組に限定することにより，同じルールに同じga組を重複して代入することを防いでいる．また，$\Delta I = \phi$の場合，whileループから抜けだしてgaの生成を終了する．これは次の理由による．$\Delta I = \phi$の場合，Iには新しいgaが含まれていないので，Iのgaを使って新しいga組を作ることはできない．それ故，たとえ，Iのgaよりなるga組を作ってルールの本体に代入しても，これ以上，新しいgaを生成することはできない．すなわち，この時点で，Iの中には意味IMPがすべて計算されているからである（$I = D \cup IMP$）．

セミナイーブ法の計算量について説明する．セミナイーブ法の時間量のほとんどは$\cup_{r \in P} Eval(r, \Delta I, I)$の実行に要する時間である．すなわち，各ルール$r$に対し，ファクトおよび/または生成されたgaよりなるga組で，ルールrの本体に代入可能なga組を重複したり洩れたりすることがないように計算し，そして，それをルールrの本体に代入して頭部についてのgaを生成する時間である．そのようなga組の計算は，通常，大雑把には，そのga組の数程度の時間で行うことができる．また，ルールの本体にga組を一つ代入して頭部についてのgaを生成する時間は定数時間である．故に，「セミナイーブ法の時間量はセミナイーブ法の実行中に各ルールの本体に代入されるga組の総数である」（仮定1と呼ぶ）とすることが多い[BR86, SPS87]．また，ルールの本体にga組が代入されるごとに，頭部のgaが一つ生成されるので，仮定1を言い換えて，「セミナイーブ法の時間量は生成されるgaの総数（但し，重複して生成されるgaも数に含める）である」（仮定2と呼ぶ）とすることもある．セミナイーブ法の領域量については，セミナイーブ法が生成した異なるgaをすべてIに記憶するために，「セミナイーブ法の領域量は生成される異なるgaの総数である」（性質1と呼ぶ）．本論文においても，セミナイーブ法の計算量を評価する際には，仮定1または2，および，性質1を使う．

なお，セミナイーブ法は他の問い合わせ処理法の中で使われることが多く，そのような問い合わせ処理法の計算量のほとんどはセミナイーブ法の実行に要する計算量である．故に，そのような問い合わせ処理法の計算量を評価する際にも，仮定1または2，および，性質1を使う．

セミナイーブ法の計算量の例を示す．例1.1の問題$(sg(b, Y)?, (P1, D1))$において，ルール集合$P1$と問い合わせアトム$sg(b, Y)?$をこのように固定し，ファクト集合$D1$を変化させて様々な問題を作り，これらの問題に対するセミナイーブ法の最悪計算量を評価する．ファクト集合中の異なる人の数をn，Parファクトの数をeと記す．このとき，ルー

ル(1.1)の本体$Person(X)$に代入されるga組の数はnである．また，ルール(1.2)の本体$Par(X_p, X) \wedge sg(X_p, Y_p) \wedge Par(Y_p, Y)$では，$Par(X_p, X)$アトムと$Par(Y_p, Y)$アトムに代入される二つの$Par$ファクトが定まると，$sg(X_p, Y_p)$アトムに代入可能な$sg$ gaも定まるので，Parファクトの数がeであり，かつ，代入可能な適切なsg gaが必ずしも生成されないことも考慮して，ルール(1.2)の本体に代入されるga組の数はe^2以下である．従って，二つのルールに代入されるga組の総数は$(n + e^2)$以下であり，$n \leq e$より，例1.1の問題に対するセミナイーブ法の最悪時間量は$O(e^2)$になる．また，生成される異なるsg gaの数はn^2以下であるので，セミナイーブ法の最悪領域量は$O(n^2)$になる．

1.4.2 同世代問題を効率的に解くための問い合わせ処理法の概要

演繹データベースの分野における有名な問題の一つに，再帰述語の引数の数（mと記す）が2の場合の**同世代問題**（same generation problem）がある．本節では，この$m = 2$の場合の同世代問題について説明する（第2章では$m(\geq 2)$が任意の整数である場合を扱う）．

第1.1節の例1.1の問題$(sg(b, Y)?, (P1, D1))$は$m = 2$の場合の同世代問題の1例であるが，一般には，$m = 2$の場合の同世代問題は次の条件を満足する問題である．

- 問い合わせアトム（$sg(b, Y)?$）はいずれかの引数が定数である．
- IDB述語は，問い合わせアトムと同じ述語一つだけ（sgだけ）である．
- 再帰ルールは一つだけ（ルール(1.2)だけ）であり，次の形をしている．

$$sg(X, Y) \leftarrow A(X_p, X), B(Y_p, Y), sg(X_p, Y_p).$$

ここで，A, BはEDB述語であり，変数X, Y, X_p, Y_pはすべて異なる．

同世代問題はdatalog問題であり，また，線形問題（すなわち，すべてのルールが線形である問題）でもある．

$m = 2$の場合の同世代問題を効率的に解くことを目的として，数多くの問い合わせ処理法が提案されている．有名な方法には**計数法**（counting method）[BMSU86]や**HaNa法**（HaNa method）[HN88, HN91]がある．以下，これら二つの方法の概要を紹介する．なお，以下の説明では簡単のため，$m = 2$の場合という形容を省略する．

(1) **計数法**

(1.1) **問題の解き方**

問題のルール集合および問い合わせアトムを例1.1のそれらとする．すなわち，問題を$(sg(b, Y)?, (P1, D))$とする．一般に，問い合わせ処理の効率化にとって，生成するga集合（S_IMPと記す）の大きさを小さくすることが重要である．計数法は，$|S_IMP|$を小さくするために，与えられた問題のIDB述語sgのgaを生成するかわりに，二つの新しいIDB述語（u_sg, d_sgと記す）のgaを生成し，そこから解集合を得る．ここで，述語u_sg, d_sgの引数のとりうる値の数は述語sgのそれよりも小さいため，生成されるu_sg ga，d_sg gaの数は生成されるsg gaの数より少なくなる．

具体的に説明する．初め，計数法は同世代問題のルール集合$P1$を書き換えて，述語u_sgとd_sgを含む新しいルール集合P'を作る．

$$P': \quad u\_sg(b, 0) \leftarrow . \tag{1.14}$$

$$u\_sg(X_p, J) \leftarrow Par(X_p, X), u\_sg(X, I), J = I + 1. \tag{1.15}$$

$$d\_sg(X, I) \leftarrow u\_sg(X, I), Person(X). \tag{1.16}$$

$$d\_sg(Y, I) \leftarrow Par(Y_p, Y), d\_sg(Y_p, J), I = J - 1, I \geq 0. \tag{1.17}$$

$$sg(b, Y) \leftarrow d\_sg(Y, 0). \tag{1.18}$$

ルール(1.14)と(1.15)は述語u_sgを定義するためのルールであり，$u_sg(X, J)$は「人Xが問い合わせアトム$sg(b, Y)$?中の人bのJ世代上の先祖である」ことを表す．また，ルール(1.16)と(1.17)は，述語u_sgを使って，述語d_sgを定義するためのルールであり，$d_sg(Y, I)$は「人Yと人bが，$I = (b$とcの世代差$) - (Y$とcの世代差$)$であるようなある共通先祖cをもつ」ことを表す．ルール(1.18)は述語d_sgより解を求めるためのルールである（これにより正しい解が得られることは後で説明する）．述語u_sg, d_sgの第2引数は世代差であり，一方，述語sgの第2引数は人であるので，上に述べたように，前者の取りうる値の数が後者の取りうる値の数より少なくなっていることに注意して欲しい．

次に，計数法は同世代問題$(sg(b, Y)?, (P1, D))$のルール集合$P1$を上のルール集合P'に変更して新しい問題$(sg(b, Y)?, (P', D))$を作り，この新しい問題をセミナイーブ法を使って解く．

(1.2) **適用範囲**

Parファクト集合より次のようにグラフGを作る．Parファクト集合に現れる異なる人をGの節点とし，ファクト$Par(b', a')$が存在するとき，Gの節点b'から節点a'へ枝を作る．

例えば，Parファクト集合が例1.1の$D1$（式(1.3)）である場合，Gは図1.1のグラフ$G1$になる．計数法は，このグラフGが閉路を持たないときのみ，同世代問題に適用可能である．

Gが閉路を含む場合に適用できないのは次の理由による．$u_sg(X, I)$の第2引数I（すなわち，bとXの世代差）は，グラフGにおける節点Xから節点bへの路の長さである．Gが閉路を含む場合，この路の長さは無限個の異なる値をとる場合がある．同様に，$d_sg(Y, J)$の第2引数Jも無限個の異なる値をとる場合がある．すなわち，新しい問題にセミナイーブ法を適用した場合，無限個のu_sg gaやd_sg gaが生成される場合があるからである．

(1.3) **正当性**

グラフGは閉路を含まないと仮定する．同世代問題のルール集合$P1$および問い合わせアトム$sg(b, Y)?$から分かるように，あるga（$sg(b, d)$と記す）が同世代問題の解であるための必要十分条件は，「bとdが，ファクト$Person(c)$が存在し，かつ，(bとcの世代差) = (dとcの世代差)であるようなある共通先祖cをもつ」ことである．同様に，新しい問題のルール集合P'および問い合わせアトム$sg(b, Y)?$から分かるように，あるga $sg(b, d)$が新しい問題の解であるための必要十分条件も上と同じである．故に，同世代問題の解集合と新しい問題の解集合は等しい．

(1.4) **効率**

計数法とセミナイーブ法の効率を比較する．ファクト集合中の異なる人の数をn，Parファクトの数をeと記す．また，Parファクト集合が表す親子関係における最大世代差（すなわち，グラフGにおける最長路の長さ）をhと記す（$h \leq n \leq e$）．計数法の時間量のほとんどはルール(1.15)と(1.17)の処理に要する時間である．ルール(1.15)の本体に代入可能なParファクトの数はe以下である．また，本体にParファクトを代入したとき，$u_sg(X, I)$のXの値が定まるので，そのとき，$u_sg(X, I)$に代入可能なu_sg gaの数は，Iの取りうる値がh以下であるので，h以下である．故に，計数法の実行中にルール(1.15)の本体に代入されるga組の数はeh以下である．同様に，ルール(1.17)の本体に代入されるga組の数もeh以下である．従って，計数法の時間量は$O(he)$である．$h \leq n$であるので，計数法の最悪時間量は$O(ne)$になる．一方，セミナイーブ法の最悪時間量は，第1.4.1節に示したように，$O(e^2)$である．(a) $n \leq e$であるので，計数法の最悪時間量$O(ne)$はセミナイーブ法の最悪時間量$O(e^2)$より小さい．また，(b) hが小さいファクト集合をもつ問題では，計数法の時間量$O(he)$は大変小さくなるので，計数法の時間量はセミナイーブ法のそれより大変小さい．

(2) **HaNa法**

HaNa法は第2章において詳しく紹介するので，ここでは，HaNa法の特徴や計算量についての結果だけを簡単に紹介する．HaNa法は，Gが閉路を含む同世代問題も解けるように，計数法を拡張した方法である．HaNa法は，u_sg ga，d_sg gaの第2引数に入れるデータとして，Gの路の長さの値ではなく，Gの路の長さを表す式（**簡略化距離集合**（cyclic simplified distance set）と呼ばれる）を計算する．この部分にHaNa法の工夫が凝らされている．この式そのものの説明は，容易ではないので，第2章において行うが，この式は次の性質をもつ．仮に，Gが閉路を含む同世代問題に計数法を適用した場合に，計数法がu_sg ga，d_sg gaの第1引数の各値に対して計算する第2引数の無限個の値（無限個のGの路の長さ）を，この式は有限（$O(n)$）の長さの一つの式で表すことができる．その結果，Gが閉路を含む同世代問題に対しても，計数法のように無限個のgaを生成することなく，HaNa法は有限個のga相当データ（HaNa法が生成するgaは第2引数の値が式であるので，正確にはgaではない．故に，ga相当データと呼ぶ）を生成し，そこから解集合を得ることができる．なお，HaNa法は，計数法とは異なり，セミナイーブ法を利用しない．

HaNa法とセミナイーブ法の最悪時間量を比較する．問題は，計数法のときと同じく，$(sg(b, Y)?, (P1, D))$とする．但し，Parファクト集合を表すグラフGは閉路を含んでもよい．HaNa法の最悪時間量は，Gにおける閉路の有無に関わらず，$O(ne)$である[HN88, HN91]．一方，セミナイーブ法のそれは$O(e^2)$であった．$n \leq e$であるので，HaNa法の最悪時間量$O(ne)$はセミナイーブ法の最悪時間量$O(e^2)$より小さい（例1.1の問題は，計数法またはHaNa法を使うと，このように効率的に解くことができる）．

1.4.3 一般の問題を効率的に解くための問い合わせ処理法の概要

一般の問題を効率的に解くことを目的とした問い合わせ処理法が数多く提案されている．これら一般の問題のための問い合わせ処理法は，$m = 2$の同世代問題のための計数法，HaNa法（第1.4.2章）や，右線形問題（right linear problem）のためのNRSU法[NRSU89b]ほどは効率的ではない．しかし，同世代問題や右線形問題のような，大変効率的な問い合わせ処理法が知られている問題は少ないので，それ以外の多くの問題を解くときは，これら一般の問題のための問い合わせ処理法が必要である．

第1.4.2節に述べたように，問い合わせ処理の効率化にとって，生成するga集合S_IMPの大きさを小さくすることが重要である．一般の問題のための方法はいずれも制約を求め，

制約を使って解の生成に無関係なgaの生成をできるだけ防ぐことで，S_IMPを小さくしている．

また，問い合わせ処理の効率化にとって，計算の重複を防ぐことも大切である．大きなルールを補助的な述語を導入して複数の小さなルールに分割し，大きなルールの各部分の計算結果を補助的な述語のgaとして記憶することにより，記憶量の増加と引き替えに，計算の重複を防ぐことができる場合がある（ただし，記憶量が爆発する場合があるので，このような場合には使えない）．そのため，一般の問題のための方法の一部は制約の利用に加えてルールの分割も併用している．

一般の問題を効率的に解くための有名な方法に，**基本マジック集合法**（basic magic sets method）[BMSU86]，**一般化マジック集合法**（generalized magic sets method）[BR87]，**一般化補助マジック集合法**（generalized supplementary magic sets method）[BR87]，**マジックテンプレート法**（magic templates method）[Ram88]がある．これらの方法は総称して**マジック集合法**（magic sets method）と呼ばれている．また，その他の有名な方法に，**アレクサンダー法**（Alexander method）[RLK86]，**アレクサンダーテンプレート法**（Alexander templates method）[Sek89]，**HCT/R法**（horn clause transformation by restrictor）[MYHI89]がある．できるだけ広い範囲の問題に対して優れた制約を作れるように，基本マジック集合法を一般化したものが一般化マジック集合法と一般化補助マジック集合法であり，更に，一般化マジック集合法を一般化したものがマジックテンプレート法である．同様に，アレクサンダー法を一般化したものがアレクサンダーテンプレート法である．一般化補助マジック集合法とアレクサンダー法とアレクサンダーテンプレート法は制約の利用に加えてルールの分割も併用している．マジックテンプレート法とアレクサンダーテンプレート法とHCT/R法は，ルール分割を除いて，同じ能力をもっている．これら三つの方法は，一般の問題を効率的に解くための従来の方法の中で最も適用範囲が広く，datalog問題クラスと一部のホーン問題を解くことができる．また，これら三つの方法は，一般の問題を効率的に解くための従来の方法の中で，最も優れた制約を作れるという意味で，最も効率的である．なお，いずれの方法に対してもルール分割を導入したり削除したりすることは容易であるので，ルール分割の使用の有無を等しくすれば，これら三つの方法の効率は完全に等しくなる．

以下では，初めに，制約のみを用いルール分割を用いないマジック集合法（すなわち，基本マジック集合法，一般化マジック集合法，マジックテンプレート法）について説明する．説明は，問題の解き方（1），正当性（2），効率（3）に分けて行う．次に，ルール分割も

併用するマジック集合法（すなわち，一般化補助マジック集合法）について，他のマジック集合法との相違点を述べることにより，説明する（4）．アレクサンダー法，アレクサンダーテンプレート法，HCT/R法については陽には説明しないが，これらの方法もマジック集合法の説明（1，2，3，4）とほぼ同じである．

（1）問題の解き方

基本マジック集合法，一般化マジック集合法，マジックテンプレート法を，(1)，(2)，(3)を通して，簡単ため，単に，マジック集合法と呼ぶ．

意味IMPの中のgaで解の生成に使われるga（解であるgaも含む）の集合は**関連集合**（relevant set）と呼ばれる．関連集合をRELと記すと，$ANS \subseteq REL \subseteq IMP$である．関連集合を包含するgaの集合は**潜在的関連集合**（potentially relevant set）と呼ばれる．潜在的関連集合を$PREL$と記すと，$REL \subseteq PREL$である．$PREL$はIMPに包含されていなくてもよい．マジック集合法は，(i) 初め，問い合わせ中の拘束をルールに伝達することにより，ある小さなga集合（MSと記す）を生成し，そのMSを使って$PREL$を定義する．このga集合MSは**マジック集合**（magic set）と呼ばれる．(ii) 次に，$PREL$を制約として使って，$PREL$に含まれるgaの生成のみを許しながら，与えられた問題の演繹データベースSからgaを生成していく．このとき，ga集合$IMP \cap PREL$が生成される．(iii) 最後に，生成されたga集合$IMP \cap PREL$の中から解集合ANSを選択する．ここで，マジック集合法が生成するga集合S_IMPは$S_IMP = MS \cup (IMP \cap PREL)$である．

問題を第1.1節の例1.1の$(sg(b,Y)?, (P1, D1))$として，マジック集合$MS1$および潜在的関連集合$PREL1$を具体的に示す．問い合わせ$sg(b,Y)?$中の定数bをルール(1.2)の頭部アトム$sg(X,Y)$の第1引数Xに伝達し，そのbを本体の$Par(X_p, X)$を経由して本体の$sg(X_p, Y_p)$の第1引数X_pに伝達すると，X_pにはbの親（b'と記す）が伝達される．更に，そのb'をルール(1.2)の頭部$sg(X,Y)$の第1引数Xに伝達し，再び，そのb'を$Par(X_p, X)$を経由して$sg(X_p, Y_p)$の第1引数X_pに伝達すると，X_pにはb'の親，すなわち，bの祖父母（b''と記す）が伝達される．マジック集合法は，このような値の伝達を繰り返すことにより，sgアトムの第1引数に伝達される値の集合として，bの先祖（b自身も含む）の集合を求める．このbの先祖の集合を表すga集合が$MS1$である．すなわち，述語を例えばm_sgと記すと（ここで，マジック集合を表すために用いたこの述語は**マジック述語**（magic predicate）と呼ばれる），$MS1 = \{m_sg(b), m_sg(b'), m_sg(b''), \ldots\}$である．また，マジック集合法は，この$MS1$を使って，第1引数の値が$MS1$の中のgaの引数の値（すなわち，$b$の先祖）であり，

第2引数の値が任意の人であるようなsg gaの集合を$PREL1$ （$=\{sg(b^\dagger, c^\dagger) \mid m\_sg(b^\dagger) \in MS1$、$c^\dagger$は任意の人$\}$）と定める．

なお，上の説明において，アトムの引数に伝達される値は**拘束**（binding）と呼ばれる．また，拘束を伝達する経路は**横方向情報伝達戦略**（sideways information passing strategy, sipと略す）と呼ばれる[BR87. Ram88]．上の例では拘束の伝達経路は一つだけであるが，一般には複数有る．このような場合，どの経路を選択するかにより得られる拘束の集合が異なるので，このように呼ばれる．

マジック集合法は，具体的には，上に述べた計算(i),(ii),(iii)を，第1.4.2節の計数法と同じく，初め，問題（$(q(\boldsymbol{b}, \boldsymbol{Y})?, (P, D))$と記す）のルール集合$P$を書き換えて新しいルール集合$P^{mg}$を作り，次に，問題の$P$を$P^{mg}$に変更してできる新しい問題$(q(\boldsymbol{b}, \boldsymbol{Y})?, (P^{mg}, D))$をセミナイーブ法を使って解くことにより行う．

上の例と同じ問題$(sg(b, Y)?, (P1, D1))$に対してマジック集合法が作る新しいルール集合$P1^{mg}$を示す．

$P1^{mg}$:

$$m\_sg(b) \leftarrow . \tag{1.19}$$

$$m\_sg(X_p) \leftarrow m\_sg(X), Par(X_p, X). \tag{1.20}$$

$$sg(X, X) \leftarrow m\_sg(X), Person(X). \tag{1.21}$$

$$sg(X, Y) \leftarrow m\_sg(X), Par(X_p, X), sg(X_p, Y_p), Par(Y_p, Y). \tag{1.22}$$

ルール(1.19)と(1.20)はマジック集合$MS1$を求めるためのルールであり，ルール(1.19)は問い合わせアトム$sg(b, Y)?$中の定数bから作られており，また，ルール(1.20)は，上に説明した，ルール(1.2)における拘束の伝達を模して作られている．これらのルールは**マジックルール**（magic rule）と呼ばれる．また，ルール(1.21)と(1.22)は$PREL1$に属すsg gaのみに限定しながら演繹データベース$(P1, D1)$よりsg gaを生成するためのルール，すなわち，ga集合$PREL1 \cap IMP1$を生成するためのルールであり，ルール(1.21)はルール(1.1)の本体に，また，ルール(1.22)はルール(1.2)の本体に，各々，マジック述語のアトムを付加して作られている．これらのルールは**変形ルール**（modified rule）と呼ばれる．

(2) **正当性**

IMPおよび$PREL$の性質より$(PREL \cap IMP) \supseteq REL \supseteq ANS$が成立する．そのため，$IMP \cap PREL$の中から解を選択しても，解集合$ANS$を正しく求めることができる．

(3) **効率**

問題を任意の問題とする．但し，ルール集合は任意でよいが，ファクト集合については，よくあるケースとして，ファクトはほぼ一様に分布しており（仮定1），かつ，あまり密には存在しない（仮定2）と仮定する．このような問題に対するマジック集合法とセミナイーブ法の効率を比較する．マジック集合法の計算時間（T_{ms}と記す）はga集合MSを生成する時間（T'_{ms}と記す）とga集合$IMP \cap PREL$を生成する時間（T''_{ms}と記す）の和である（$T_{ms} = T'_{ms} + T''_{ms}$）．仮定1が成立する場合，$T''_{ms}$とセミナイーブ法の計算時間（$T_{sn}$と記す）の比は，ga集合$IMP \cap PREL$とga集合$IMP$の大きさの比にほぼ等しい（$T''_{ms}/T_{sn} \doteq |IMP \cap PREL|/|IMP|$）．従って，マジック集合法の計算時間は

$$T_{ms} \doteq T'_{ms} + |IMP \cap PREL|/|IMP| \times T_{sn} \tag{1.23}$$

と表すことができる．また，仮定2が成立する場合，問題の関連集合RELの大きさはIMPの大きさに比べ大変小さい（$|REL| << |IMP|$）．あまり手間をかけずに求めた（すなわち，計算時間T'_{ms}の小さいMSによって定義される）$PREL$で，RELをうまく近似する（すなわち，$IMP \cap PREL \doteq REL$である）$PREL$は優れている，と言う．$PREL$が優れていれば，T'_{ms}が小さく，かつ，$|IMP \cap PREL| \doteq |REL|$であるので，マジック集合法の計算時間$T_{ms}$（式(1.23)）は$|REL|/|IMP| \times T_{sn}$に近づき，そして，この$|REL|/|IMP| \times T_{sn}$は，$|REL| << |IMP|$より，$T_{sn}$に比べ大変小さい．このように，仮定1，2のファクト集合をもつ問題に対しては，$PREL$が優れていればその程度に応じて，マジック集合法の計算時間T_{ms}はセミナイーブ法の計算時間T_{sn}より小さくなる．

問題を再び任意の問題とする．但し，上記の場合と異なり，ファクト集合についても仮定を設けない．ルール集合を固定しファクト集合を変化させて様々な問題を作り，マジック集合法の最悪時間量とセミナイーブ法のそれを比較する．マジック集合法はT'_{ms}が小さくなるようにマジック集合MSを作り，また，$T''_{ms} \leq T_{sn}$である．従って，任意の問題に対し，マジック集合法の計算時間T_{ms}はセミナイーブ法の計算時間T_{sn}とほぼ等しいかまたはそれより小さい（性質1）．ところで，セミナイーブ法の最悪時間量を与えるファクト集合をもった問題の中に，意味IMPと関連集合RELが一致するような（$REL = IMP$）問題があると仮定する（仮定3）．この問題に対しマジック集合法を適用しても，$REL = IMP$かつ$REL \subseteq (IMP \cap PREL) \subseteq IMP$であるために$(IMP \cap PREL) = IMP$となり，$PREL$は制約として機能しない．この場合，$T''_{ms}$は$T_{sn}$と等しくなる．故に，この問題に対するマジック集合法の計算時間T_{ms}は，小さな計算時間T'_{ms}を無視すれば，セミナイーブ法の最

悪時間量に一致する（性質2）．性質1，2より，仮定3の問題がある場合，マジック集合法の最悪時間量はセミナイーブ法のそれにほぼ等しくなる．

(4) **一般化補助マジック集合法**

一般化補助マジック集合法は$PREL$の利用以外にルール分割も併用するので，上記の(1)から(3)とは少し異なる点がある．一般化補助マジック集合法と他のマジック集合法との主な相違点を述べる．

一般化補助マジック集合法が生成するルール集合$P^{mg'}$は，他のマジック集合法が生成するルール集合P^{mg}の中のルールを分割したようなルール集合になる．例えば，上の例と同じ問題$(sg(b,Y)?,(P1,D1))$に対して一般化補助マジック集合法が作るルール集合$P1^{mg'}$は次のようになる．

$$P1^{mg'}: \quad m\_sg(b) \leftarrow . \tag{1.24}$$

$$sup_1(X,X_p) \leftarrow m\_sg(X), Par(X_p,X). \tag{1.25}$$

$$sup_2(X,Y_p) \leftarrow sup_1(X,X_p), sg(X_p,Y_p). \tag{1.26}$$

$$m\_sg(X_p) \leftarrow sup_1(X,X_p). \tag{1.27}$$

$$sg(X,X) \leftarrow m\_sg(X), Person(X). \tag{1.28}$$

$$sg(X,Y) \leftarrow sup_2(X,Y_p), Par(Y_p,Y). \tag{1.29}$$

ルール集合$P1^{mg}$と$P1^{mg'}$を比べると，$P1^{mg}$の中のルール(1.22)が，$P1^{mg'}$では，補助的な述語sup_1とsup_2を使って三つのルール(1.25)，(1.26)，(1.29)に分割されているのが分かる．述語sup_1とsup_2は**補助マジック述語**（supplementary magic predicate）と呼ばれる．

一般化補助マジック集合法と他のマジック集合法の領域量，時間量を比較した場合，一般に，前者の領域量は補助マジック述語のgaを記憶するために後者のそれより大きく，また，前者の時間量は後者のそれと同程度かまたはそれより小さい．

ある種の問題では，一般化補助マジック集合法の領域量が他のマジック集合法のそれよりあまり増加しないのに，前者の時間量は後者のそれより大幅に減少することがある．このような場合，一般化補助マジック集合法は他のマジック集合法より効率的である．

例えば，$m=2$の同世代問題に対しては，一般化補助マジック集合法の最悪領域量も他のマジック集合法のそれもともに$O(n^2)$である．しかし，他のマジック集合法の最悪時間量が，先に述べたように，セミナイーブ法の最悪時間量$O(e^2)$と等しいのに対し，一般化補助マジック集合法の最悪時間量は$O(ne)$となる．この問題に対しては，一般化補助マジック集合法は他のマジック集合法より効率的である．

なお，$m=2$の同世代問題に対する一般化補助マジック集合法の最悪時間量$O(ne)$は，計数法およびHaNa法の最悪時間量と等しい．

1.5 本研究の概要

演繹データベースシステムにおける再帰的な問い合わせ処理の効率化を目的として，第1.4節で紹介した方法等，すでに多くの問い合わせ処理法が提案されている．しかし，問い合わせ処理の効率化のためには，未だ十分ではないように思われる．

本研究では，再帰的な問い合わせ処理の効率化を更に進めるために，三つの新しい問い合わせ処理法（拡張HaNa法，直積法，緩和法と呼ぶ）を提案する．いずれの方法も，ホーン問題クラスのある部分クラスを対象としている．拡張HaNa法は適用範囲は狭いが大変効率的であり，一方，緩和法は拡張HaNa法ほどは効率的でないが適用範囲は広い．直積法は両者の中間的な適用範囲と効率をもつ．本研究では，また，直積法の内部処理に関連して基本的な組合せ問題（多次元直方体被覆問題と呼ぶ）を考え，この問題を効率的に解くためのアルゴリズムも提案する．このアルゴリズムを組み込んだ直積法は，適用範囲は狭くなるが，しかし，より効率的に問い合わせを処理することができる．

拡張HaNa法（文献[SIK92, SIK93]）について説明する．ファクト集合が表すグラフGが閉路を含んでよく，かつ，再帰述語sgの引数の数mが一般の整数（$m \geq 2$）である場合の同世代問題を考える．$m=2$の同世代問題については多くの研究が行われ，第1.4.2節および第1.4.3節で紹介したように，HaNa法やマジック集合法等の最悪時間量が解析されている．$m=2$の場合，従来の方法の中で，最悪時間量において，HaNa法およびマジック集合法（但し，一般化補助マジック集合法）が最も効率的であった．一方，$m \geq 3$の同世代問題についてはあまり研究が行われていない．$m \geq 3$の場合，従来の方法の中で，最悪時間量において，マジック集合法（但し，一般化補助マジック集合法）が最も効率的であると思われる．本研究では，$m \geq 3$の同世代問題がより効率的に解けるようにHaNa法を変形して**拡張HaNa法**（modified HaNa method）を提案する．拡張HaNa法は，効率化のために，Gの路の長さを表すためのHaNa法の式を多数まとめて扱う，等の工夫をしている．拡張HaNa法の最悪時間量を解析し，$m \geq 3$の場合，通常よく生じるデータに対して拡張

HaNa法がマジック集合法より効率的であることを示す．拡張HaNa法は第2章で詳しく説明する．

直積法（文献[SIK90]の一般化）について説明する．datalog問題クラスの新しい部分クラスを定義し，この問題クラスを**直積問題**（Cartesian product problem）クラスと呼ぶ．直積問題クラスは第1.4.2節の同世代問題クラスを包含し，かつ，一部の非線形問題（一つ以上の非線形ルールを含む問題を非線形問題と言う）も含む．直積問題クラスのための問い合わせ処理法として**直積法**（Cartesian product method）を提案する．pをm引数をもつ述語，$C_1,\ldots,C_m$の各々を定数の集合とするとき，gaの集合$\{p(a_1,\ldots,a_m) \mid a_1 \in C_1,\ldots,a_m \in C_m\}$を$p[C_1\times\cdots\times C_m]$と記し，この式$p[C_1\times\cdots\times C_m]$を**直積型基礎アトム集合式**（ground atom set expression of Cartesian product type），略して，**gase**と呼ぶ（より一般的には，C_iは定数組の集合でよい．詳しくは第3章を参照）．直積問題では，gaの代わりにgaseを生成し，意味IMPをそれらgaseの和として表すことができる．定義から分かるように，gaseは一つのgaseで多数のgaを表すことができるので，少数のgaseを生成するだけでIMPを表せる可能性がある．一つのgaseを生成し処理するのに要する時間は一つのgaのそれに比べ大きいが，もし，gaseの数が大変小さいならば，このようにgaseを生成し，そこから解集合ANSを得ることにより，問い合わせ処理の計算時間を全体として減らせる可能性がある．直積法は，この考えに基づいており，gaseを生成し，そこから解集合ANSを得る．直積問題クラスは比較的広い問題クラスであるので，従来の方法の中で直積問題クラスに適用できるのは一般の問題を対象とした方法だけと思われる．直積問題クラスに対し，従来の方法の中で，マジック集合法が最も効率的であると思われる．直積問題の例として，$m=3$の場合の同世代問題の再帰ルールを複数にした問題，および，$m=2$の場合の同世代問題の再帰ルールを非線形にした問題を考え，これらの問題を使って，直積法とマジック集合法の効率を比較する．最悪時間量においては，直積法はマジック集合法より非効率である．しかし，計算機実験によって，ファクト集合がファクトを密に含む問題に対しては，直積法がマジック集合法より効率的であることを示す．直積法は第3章で詳しく説明する．

多次元直方体被覆問題を解くアルゴリズムを説明するための準備として，直積法について補足する．直積法は，gaseを一つ生成するたびに，そのgaseが表すga集合の中に新しいgaが含まれているか否かを検査し，新しいgaが含まれている場合はそのgaseを保存し，そうでない場合はそのgaseを捨てる．新しいgaが含まれているか否かの検査は次のように行う．今生成したgaseを$p[C_{01}\times\cdots\times C_{0m}]$と記し，また，それまでに生成し保存しているgaseの

中で，$p[C_{01}\times\cdots\times C_{0m}]$と同じ述語$p$をもつgaseを$p[C_{11}\times\cdots\times C_{1m}],\ldots,p[C_{n1}\times\cdots\times C_{nm}]$と記す．直積法は，式

$$C_{01}\times\cdots\times C_{0m} \subseteq C_{11}\times\cdots\times C_{1m}\cup\cdots\cup C_{n1}\times\cdots\times C_{nm} \tag{1.30}$$

が成立するか否かを検査し，式(1.30)が成立しない場合，gase $p[C_{01}\times\cdots\times C_{0m}]$の中に新しいgaが含まれていると判定し，一方，式(1.30)が成立する場合，新しいgaは含まれていないと判定する．直積法の計算時間のほとんどはこの検査，すなわち，式(1.30)の判定に要する時間である．

直積問題クラスの部分クラスを定義する．ファクトの引数は整数であり，かつ，ファクト集合は“連続である”とする（ファクト集合の連続の定義は第4章で述べる）．また，ルールの形も少し制限する．直積問題クラスをこのように制限してできる問題クラスを**連続直積問題**（continuous Cartesian product problem）クラスと呼ぶ．連続直積問題に直積法を適用した場合，生成されるgase $p[C_1\times\cdots\times C_m]$の中の各集合$C_j, j=1,\ldots,m,$は連続整数集合となる．すなわち，$C_j$はある整数$b_j$と$e_j$を使って，$C_j=\{x_j \mid b_j\le x_j\le e_j,\ x_j$は整数$\}$と表すことができる．その結果，式(1.30)の$C_{ij}, i=0,1,\ldots,n, j=1,\ldots,m,$も連続整数集合となる．$C_{ij}$が連続整数集合である場合，この連続性を利用することにより，式(1.30)の判定を，C_{ij}が不連続である一般の場合より，より効率的に行える可能性がある．そして，式(1.30)の判定をより効率的に行うことができれば，その判定アルゴリズムを直積法に組み込むことにより，連続直積問題に対しては，直積法の効率を更に高めることができる．なぜならば，上に述べたように，直積法の計算時間のほとんどは式(1.30)の判定に要する時間であるからである．

多次元直方体被覆問題を解くアルゴリズム（文献[SI97]）について説明する．上に述べたように，C_{ij}が連続整数集合である場合に式(1.30)を判定する問題を**多次元直方体被覆問題**（multi-dimensional rectangle cover problem）と呼ぶ．多次元直方体被覆問題は組合せ問題の分野において有名な充足可能性問題の一般化でもある．本研究では，新しいアルゴリズムを提案すること，および，充足可能性問題のための従来のアルゴリズムを多次元直方体被覆問題に素直に拡張することにより，多次元直方体被覆問題を解くための複数のアルゴリズムを与える．そして，計算機実験により，これらのアルゴリズムが多次元直方体被覆問題を効率的に解くことを示す（その結果，上に述べたように，これらのアルゴリズムを直積法に組み込むことにより，連続直積問題に対しては，直積法の効率は更に高まる）．多

次元直方体被覆問題を解くアルゴリズムは第4章で詳しく説明する．

緩和法（文献[SIK91, SIK94]）について説明する．一般の問題を効率的に解くための問い合わせ処理法として**緩和法**（relaxation method）を提案する．緩和法は，第1.4.3節で紹介したマジック集合法と類似した方法であり，マジック集合法と同様，潜在的関連集合$PREL$を求め，$PREL$を制約として使って解を得るのに必要と思われるgaのみを生成し，生成されたgaの集合$IMP \cap PREL$の中から解を選択する．しかし，緩和法はマジック集合法とは異なるやり方で$PREL$を求める．緩和法は，$PREL$は関連集合RELの緩和であるという考えに基づいて，$PREL$をより柔軟に求める．先に述べたように，あまり手間をかけずに求めた$PREL$で，RELをうまく近似する（すなわち，$IMP \cap PREL \doteq REL$である）$PREL$は優れている，と言う．緩和法やマジック集合法のように，$PREL$を利用する問い合わせ処理法では，二つの方法の効率の優劣はそれらの方法が作る$PREL$の優劣により定まる（但し，ある問題では効率化のためにルール分割を併用することもできるので，ルール分割の使用の有無は同じであると仮定する）．マジック集合法の一つであるマジックテンプレート法は，一般の問題を効率的に解くための従来の方法の中で最も適用範囲が広く，datalog問題クラスと一部のホーン問題を解くことができ，かつ，最も優れた$PREL$を作ることができる，すなわち，最も効率的である（但し，アレクサンダーテンプレート法とHCT/R法もマジックテンプレート法と同じ能力を持つ）．マジックテンプレート法と緩和法を比較し，(1) マジックテンプレート法が解ける任意の問題に対し，緩和法がマジックテンプレート法が作るのと同じ$PREL$を作れる（他の$PREL$も作れる）こと，すなわち，任意の問題を，緩和法がマジックテンプレート法と少なくとも同じ効率で解けること，(2) マジックテンプレート法が解けるある問題に対し，緩和法がマジックテンプレート法が作れない優れた$PREL$を作れること，すなわち，ある問題を緩和法がマジックテンプレート法よりも効率的に解けること，(3) マジックテンプレート法が解けないある問題を緩和法が効率的に解けること，を示す．なお，緩和法の適用範囲はdatalog問題クラスと一部のホーン問題であるが，ホーン問題部分がマジックテンプレート法より広い．緩和法は第5章で詳しく説明する．

最後に，これまで説明した問い合わせ処理法が対象とする問題クラスの包含関係を図1.3に示す．図の実線の問題クラスが，本研究で提案する問い合わせ処理法が対象とする問題クラスである．

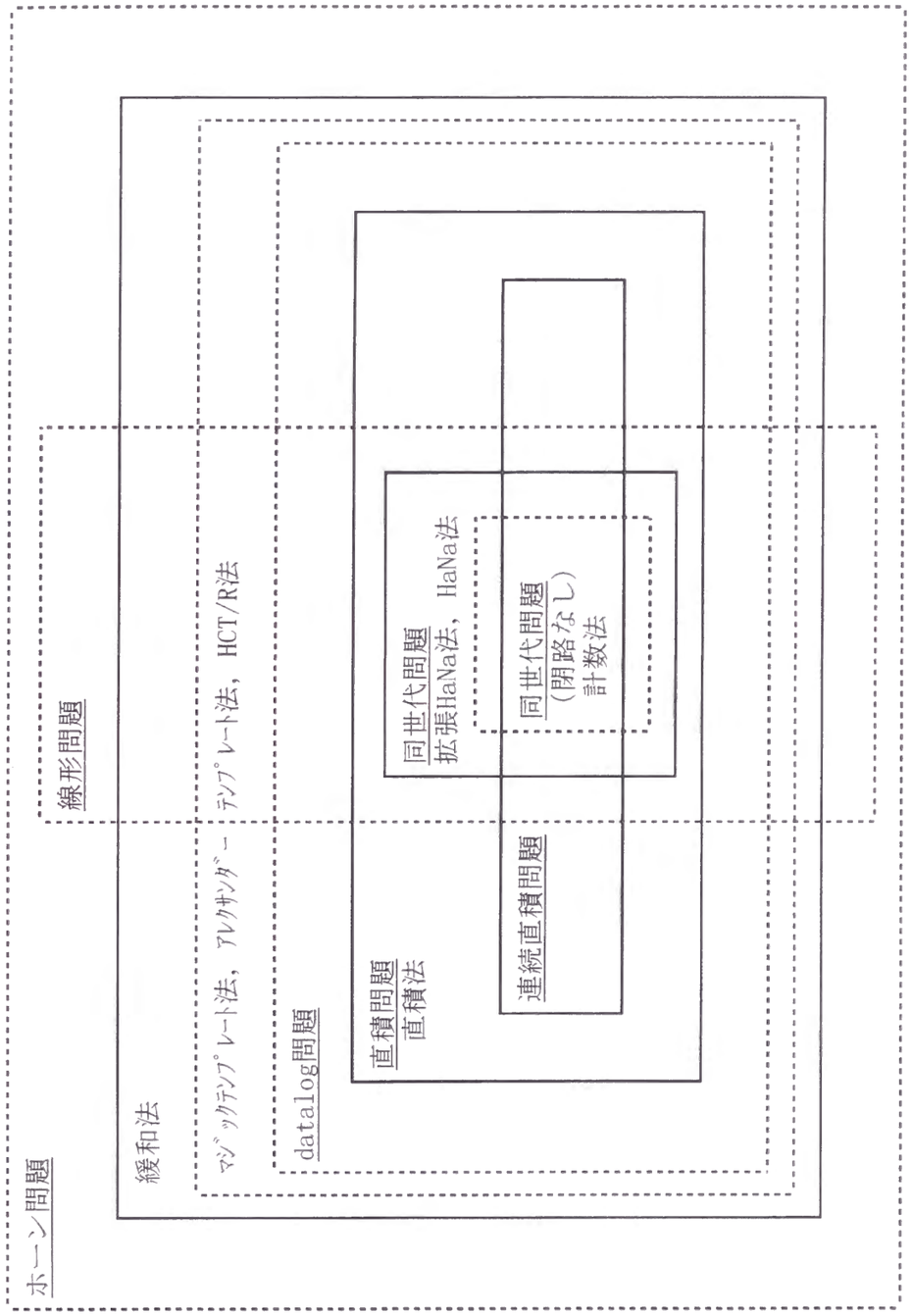


図1.3: 問題クラスの包含関係と各問題クラスのための問い合わせ処理法

第2章

多変数同世代問題を効率的に解くための問い合わせ処理法

再帰述語の引数の数mが2の場合の同世代問題は有名な問題であり，多くの研究が行われている．$m=2$の場合，問い合わせ処理法の中で，最悪時間量において，HaNa法およびマジック集合法が最も効率的である．一方，$m \geq 3$の場合，研究はあまり行われていない．$m \geq 3$の場合，マジック集合法の最悪時間量は$O(e^h n^{m-h})$となり，最悪時間量において，マジック集合法が最も効率的であると思われる．ここで，hは問い合わせアトム中の定数の数，eは再帰ルールの中の各EDB述語についてのファクトの数，nはそれら各EDB述語についてのファクトの中に現れる異なる定数の数である．本章では，$m=2$の場合に対して提案されたHaNa法を$m \geq 3$の場合に効率的になるように変形して拡張HaNa法を提案し，$m \geq 3$の場合，その最悪時間量が$O(t_0(n^m \log n + n^{m-h}|ANS| \log n))$となることを解析する．ここで，$t_0$は初期値を与えるEDB述語についてのファクトの数，$|ANS|$は解の数である．拡張HaNa法とマジック集合法を比較し，$m \geq 3$，かつ，通常の多くの問題例に見られるように，hとeが大きく，t_0と$|ANS|$が小さい場合，最悪時間量において，拡張HaNa法がマジック集合法より効率的であることを示す．

2.1 はじめに

同世代問題（same generation problem）は，ルール集合

$$P: \quad sg(X_1, X_2, \ldots, X_m) \leftarrow A_0(X_1, X_2, \ldots, X_m). \tag{2.1}$$
$$sg(X_1, X_2, \ldots, X_m) \leftarrow A_1(X_1', X_1), A_2(X_2', X_2), \ldots, A_m(X_m', X_m).$$

$$sg(X_1', X_2', \ldots, X_m'). \tag{2.2}$$

と問い合わせアトム

$$sg(a_1, a_2, \ldots, a_h, X_{h+1}, X_{h+2}, \ldots, X_m)? \tag{2.3}$$

とファクト集合Dよりなる問題$(sg(a_1, a_2, \ldots, a_h, X_{h+1}, X_{h+2}, \ldots, X_m)?, (P, D))$である．ここで，$sg$はIDB述語，$A_0$および$A_1, \ldots, A_m$はEDB述語，$a_1, \ldots, a_h$は定数，$X_1, \ldots, X_m$，$X_1', \ldots, X_m'$は変数であり，変数$X_1, \ldots, X_m, X_1', \ldots, X_m'$はすべて異なる．

なお，ファクト集合Dについては，一部の研究で設けられているような制限を設けない．すなわち，ルール(2.2)に現れる各EDB述語$A_i, i=1, \ldots, m$，に対し，ファクト集合D（$=\{A_0(c_{11}, \ldots, c_{1m}), A_0(c_{21}, \ldots, c_{2m}), \ldots\} \cup \{A_1(a_{11}', b_{11}'), A_1(a_{12}', b_{12}'), \ldots\} \cup \cdots \cup \{A_m(a_{m1}', b_{m1}'), A_m(a_{m2}', b_{m2}'), \ldots\}$）の中の$A_i$ファクトの集合$\{A_i(a_{i1}', b_{i1}'), A_i(a_{i2}', b_{i2}'), \ldots\}$から，次のように，グラフ$G_i$を作る．$A_i$ファクト集合に現れる異なる定数をグラフ$G_i$の節点とし，ファクト$A_i(a', b')$が存在するとき，$G_i$の節点$a'$から$b'$へ枝を作る．このように$G_i$を作るとき，各グラフ$G_i = (V_i, E_i)(i = 1, \ldots, m,\ V_i$：節点集合，$E_i$：枝集合)は閉路を含んでも含まなくてもよいとする．

同世代問題の解集合は，演繹データベース(P, D)から生成可能なsg ga（$sg(a_1', \ldots, a_h', a_{h+1}', \ldots, a_m')$と記す）のうち，条件$a_1' = a_1, \ldots, a_h' = a_h$を満たす全ての$sg$ gaの集合

$$\begin{aligned} ANS \ = \ & \{sg(a_1', \ldots, a_h', a_{h+1}', \ldots, a_m') \mid a_1' = a_1, \ldots, a_h' = a_h, \\ & sg(a_1', \ldots, a_h', a_{h+1}', \ldots, a_m') \text{は演繹データベース} (P, D) \text{より生成される}\} \end{aligned}$$

である．

$m=2$の場合の同世代問題は有名な問題であり，多くの研究が行われている．この問題を効率的に解くことを目的として**HaNa法**（HaNa method）[HN88, HN91]やその他多くの方法[HN84, BR86, GSS87, HH87, SZ87, AO89, HH89]が提案され，そして，それらの方法，および，より広い範囲の問題を効率的に解くことを目的とした**マジック集合法**（magic sets method）等の最悪時間量が解析されている．簡単のため，各グラフ$G_i, i=1, \ldots, m$，の大きさは同じであると仮定し，G_iの節点の数（すなわち，A_iファクト集合に現れる異なる定数の数）をn，G_iの枝の数（すなわち，A_iファクトの数）をeと記す．$m=2$の場合，例えば，HaNa法の最悪時間量は$O(ne)$[HN88, HN91]，**一般化補助マジック集合法**（generalized suplementary magic sets method）[BR87]を除くマジック集合法（すなわち，基本マジック

集合法 [BMSU86, BR86]，一般化マジック集合法 [BR87]，マジックテンプレート法 [Ram88]) の最悪時間量は $O(e^2)$ [SPS87]，一般化補助マジック集合法の最悪時間量は $O(ne)$ である．$m=2$ の場合，多くの方法が提案されているが，最悪時間量において，HaNa 法および一般化補助マジック集合法が最も優れる．

一方，$m(\geq 2)$ が一般の整数の場合，問題自身は文献 [HH89, Han89] に言及されているが，問い合わせ処理法の研究はあまり行われていない．一般の m の問題を従来の方法を使って解くには，(i) 一般の m の問題にそのまま適用可能な広い適用範囲をもつ方法 [BMSU86, SPS87, SZ86] を使って解くか，または，(ii) $m=2$ の問題に対して提案された方法 [HN88, HN91, HN84, BR86, GSS87, HH87, SZ87, AO89, HH89] を一般の m に適用できるように素直に変形し，そのようにして得られる方法を使って解けばよい．

(i) の場合，グラフ $G_1,\ldots,G_m$ の全てが閉路を含んでもよいという条件のもとでは，最悪時間量において，一般化補助マジック集合法が最も効率的であると思われる．一般化補助マジック集合法は，ルール集合 P：(2.1), (2.2), および問い合わせアトム $sg(a_1,\ldots,a_h,X_{h+1},\ldots,X_m)?$ より次の新しいルール集合

$$P^{mg}: \quad m\_sg(a_1,\ldots,a_h) \leftarrow . \tag{2.4}$$

$$m\_sup_1(X_1',X_2,\ldots,X_h) \leftarrow m\_sg(X_1,\ldots,X_h), A_1(X_1',X_1). \tag{2.5}$$

$$m\_sup_2(X_1',X_2',X_3,\ldots,X_h) \leftarrow m\_sup_1(X_1',X_2,\ldots,X_h), A_2(X_2',X_2). \tag{2.6}$$

$$\vdots$$

$$m\_sup_{h-1}(X_1',\ldots,X_{h-1}',X_h) \leftarrow m\_sup_{h-2}(X_1',\ldots,X_{h-2}',X_{h-1},X_h), A_{h-1}(X_{h-1}',X_{h-1}). \tag{2.7}$$

$$m\_sg(X_1',\ldots,X_h') \leftarrow m\_sup_{h-1}(X_1',\ldots,X_{h-1}',X_h), A_h(X_h',X_h). \tag{2.8}$$

$$sup_h(X_1,\ldots,X_h,X_{h+1}',\ldots,X_m') \leftarrow m\_sg(X_1,\ldots,X_h), A_1(X_1',X_1),\ldots,A_h(X_h',X_h), sg(X_1',\ldots,X_h',X_{h+1}',\ldots,X_m'). \tag{2.9}$$

$$sup_{h+1}(X_1,\ldots,X_{h+1},X_{h+2}',\ldots,X_m') \leftarrow A_{h+1}(X_{h+1}',X_{h+1}), sup_h(X_1,\ldots,X_{h+1}',X_{h+2}',\ldots,X_m'). \tag{2.10}$$

$$\vdots$$

$$sup_{m-1}(X_1,\ldots,X_{m-1},X_m') \leftarrow A_{m-1}(X_{m-1}',X_{m-1}), sup_{m-2}(X_1,\ldots,X_{m-2},X_{m-1}',X_m'). \tag{2.11}$$

$$sg(X_1,\ldots,X_m) \leftarrow A_m(X_m',X_m), sup_{m-1}(X_1,\ldots,X_{m-1},X_m'). \tag{2.12}$$

を作り，そして，同世代問題 $(sg(a_1,\ldots,a_h,X_{h+1},\ldots,X_m)?,(P,D))$ のルール集合 P を新しいルール集合 P^{mg} に置き換えて得られる問題 $(sg(a_1,\ldots,a_h,X_{h+1},\ldots,X_m)?,(P^{mg},D))$ をセミナイーブ法を使って解くことにより，解集合 ANS を得る．ここで，述語 m_sg は**マジック述語**（magic predicate），述語 $m\_sup_i, i=1,\ldots,h-1$, と $sup_j, j=h,\ldots,m-1$, は**補助マジック述語**（supplementary magic predicate）と呼ばれる．一般化補助マジック集合法の計算時間のほとんどはセミナイーブ法の実行に要する時間であり，そして，その時間のほとんどはルール (2.9) の処理に要する時間である．セミナイーブ法がルール (2.9) の本体に代入する ga 組の数は高々 $e^h n^{m-h}$ であるので，一般化補助マジック集合法の最悪時間量は $O(e^h n^{m-h})$ になる．

一方，(ii) の場合，述語 $sg(X_1,\ldots,X_h,X_{h+1},\ldots,X_m)$ の第 1 引数から第 h 引数までを一つの変数 Z_1（$=(X_1,\ldots,X_h)$），残りの引数をもう一つの変数 Z_2（$=(X_{h+1},\ldots,X_m)$）として，述語 sg を 2 引数述語 $sg(Z_1,Z_2)$ と見なし，また，EDB 述語の連言 $A_1(X_1',X_1)\wedge\cdots\wedge A_h(X_h',X_h)$ を変数 Z_1 に関する EDB 述語 $A_1'(Z_1',Z_1)$，同様に，$A_{h+1}(X_{h+1}',X_{h+1})\wedge\cdots\wedge A_m(X_m',X_m)$ を変数 Z_2 に関する EDB 述語 $A_2'(Z_2',Z_2)$ と見なすことにより，$m=2$ に対して提案された方法は一般の m に適用できるように素直に変形することができる．一般の m に素直に変形された方法の中で，HaNa 法は最悪時間量において最も優れていると思われる．その最悪時間量は $O(n^h(e^h+t_0)+(n^h+n^{m-h})(t_0+e^{m-h}))=O(n^h(e^h+t_0)+n^{m-h}(t_0+e^{m-h}))$ である [HN88, HN91]．ここで，n^h, e^h は，各々，述語 $A_1\wedge\cdots\wedge A_h$ が表す積グラフ $G_1\times\cdots\times G_h$ の節点数，枝数であり，n^{m-h}, e^{m-h} は，各々，述語 $A_{h+1}\wedge\cdots\wedge A_m$ が表す積グラフ $G_{h+1}\times\cdots\times G_m$ の節点数，枝数である．また，t_0 は A_0 ファクトの数である．

一般化補助マジック集合法の最悪時間量 $O(e^h n^{m-h})$ と HaNa 法の最悪時間量 $O(n^h(e^h+t_0)+n^{m-h}(t_0+e^{m-h}))$ を比べると，$m\geq 3$ の場合，従来の方法の中で，最悪時間量において，一般化補助マジック集合法が最も優れていると思われる．以後，簡単のため，一般化補助マジック集合法を単にマジック集合法と呼ぶ．

本章では，R.W.Haddad と J.F.Naughton により $m=2$ の場合に対して提案された HaNa 法 [HN88, HN91] を $m\geq 3$ の場合に効率的になるように変形して**拡張 HaNa 法**（modified HaNa method）を提案する．そして，$m\geq 3$ のとき，その最悪時間量が

$$O(t_0(n^m \log n + n^{m-h}|ANS|\log n)).$$

最悪領域量が

$$O(mnc + |ANS|)$$

であることを解析する．ここで，$|ANS|$ は解の数（$|ANS| \leq n^{m-h}$）である．$m \geq 3$，かつ，通常の多くの問題例に見られるように，h と e が大きく，t_0 と $|ANS|$ が小さい場合，最悪時間量において，拡張HaNa法がマジック集合法より優れていることを示す．また，比較的小さな領域量 $O(mnc)$ を無視すれば，拡張HaNa法の領域量がほぼ最適であることを示す．

本章では，第2.2節で m（≥ 2）が一般の整数である場合の同世代問題の応用例を示す．第2.3節でHaNa法を紹介する．第2.4節で拡張HaNa法を与え，第2.5節でその最悪計算量を解析し，第2.6節で解析結果について考察する．最後に第2.7節でまとめる．

2.2 多変数同世代問題の応用例

m 人の人が共通先祖から同じ世代数だけ下がった子孫同士である場合，それら m 人は“同世代である”と言う．親子関係を表すデータの集合（例えば，b は a の親，c は b の親，...として，$\{Par(b,a), Par(c,b), \ldots\}$ と記す）と，そこに現れる人の集合（例えば，$a, b, c, \ldots$ が人として，$\{Person(a), Person(b), Person(c), \ldots\}$ と記す）と，特定の二人の人（a_1, a_2 と記す）が与えられていて，a_1, a_2, x_3 の三人が同世代となるようなすべての人 x_3 を求めたい．この問題は $m = 3$ の同世代問題として表すことができる．以下にこれを示す（第1章の例1.1では $m = 2$ の場合について示した）．

同世代問題のルール集合を

$$P': \quad sg(X, X, X) \leftarrow Person(X). \tag{2.13}$$

$$sg(X_1, X_2, X_3) \leftarrow Par(X_1', X_1), Par(X_2', X_2), Par(X_3', X_3), sg(X_1', X_2', X_3'). \tag{2.14}$$

とし，ファクト集合を親子関係を表すデータの集合と人を表すデータの集合の和

$$D' = \{Par(b,a), Par(c,b), \ldots\} \cup \{Person(a), Person(b), Person(c), \ldots\}$$

とし，問い合わせアトムを

$$sg(a_1, a_2, X_3)?$$

とする．Par ファクト集合が定める親子関係において，x をある人とし，その x の子供を $x^i, i = 1, \ldots, d$，その x^i の子供を $x^{ij}, j = 1, \ldots, d_i$ と記す．このとき，ルール(2.13)より $sg(x,x,x)$ が得られ，その $sg(x,x,x)$ にルール(2.14)を適用すると，$sg(x^{i_1}, x^{i_2}, x^{i_3}), i_1, i_2, i_3 = 1, \ldots, d.$ が得られる．更に，その $sg(x^{i_1}, x^{i_2}, x^{i_3})$ にルール(2.14)を適用すると，$sg(x^{i_1 j_1}, x^{i_2 j_2}, x^{i_3 j_3}), j_1 = 1, \ldots, d_{i_1}, j_2 = 1, \ldots, d_{i_2}, j_3 = 1, \ldots, d_{i_3}$ が得られる．このような処理を繰り返すと，x からある同じ世代数だけ下がった任意の子孫 x_1, x_2, x_3 に対し，$sg(x_1, x_2, x_3)$ を得ることができる．すなわち，述語 $sg(X_1, X_2, X_3)$ は「X_1, X_2, X_3 の3人がある共通先祖から同じ世代数だけ下った子孫同士である」，すなわち，「X_1, X_2, X_3 の3人が同世代である」ことを示している．故に，問い合わせアトム $sg(a_1, a_2, X_3)?$ は，a_1, a_2, X_3 の3人が同世代であるようなすべての人 X_3 を見つけることを要求している．従って，確かに，三人の同世代関係を求める問題は $m = 3$ の同世代問題 $(sg(a_1, a_2, X_3)?, (P', D'))$ として表すことができる（同世代問題の名前はこのように同世代関係を求めれることに由来している）．

なお，上記の3人の同世代関係を求める問題は，$m = 2$ の同世代問題として表すことはできないし，また，幾つかの $m = 2$ の同世代問題の組み合わせとして表すことも難しいように思われる．なぜならば，$m = 2$ の同世代問題を幾つか解いて，a_1 と a_2 が同世代であることを確認し，そして，a_1 と x が同世代であり，かつ，a_2 と x も同世代であるような人 x を見つけたとしても，a_1 と a_2 と x の3人が同世代であるとは限らないからである．

故に，m（≥ 3）人の同世代関係を表せる $m \geq 3$ の同世代問題について研究することは，$m = 2$ の同世代問題が十分研究されているにも関わらず，意義があるように思われる．

2.3 HaNa法の紹介

本節では，R.W.HaddadおよびJ.F.Naughtonにより提案されたHaNa法[HN88, HN91]を紹介する．

2.3.1 HaNa法の概要

$m = 2$ の場合の同世代問題を考える．問い合わせは $sg(a_1, X_2)$? である．グラフ $G_i, i = 1, 2,$ における，節点 y_i から節点 x_i への三つの距離集合を

$D_i(y_i, x_i) = \{\ l \mid$ 節点 y_i から x_i へ長さ l の路が存在する（閉路を含んでも含まなくてもよい）$\}$,

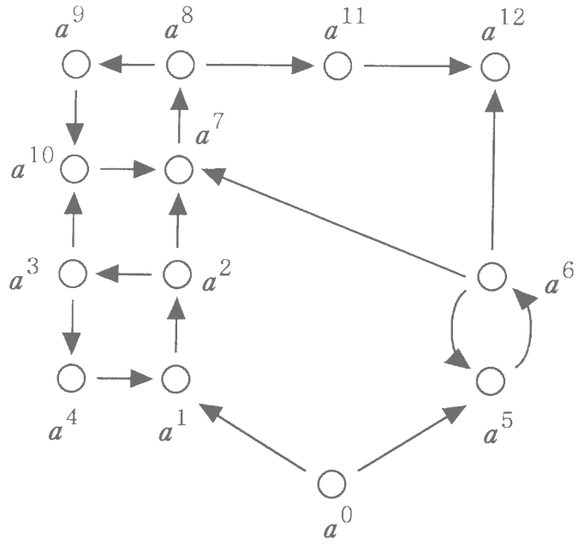


図2.1: $A_i, i = 1, \ldots, m$, ファクト集合を表すグラフG

$CD_i(y_i, x_i) = \{\, l \mid G_i$の少なくとも一つの閉路に出会う，節点$y_i$から$x_i$への長さ$l$の路が存在する $\}$,

$F_i(y_i, x_i) = \{\, l \mid$ 節点y_iからx_iへ$l < n$なる長さlの路が存在する $\}$

$= D_i(y_i, x_i) \cap \{0, 1, 2, \ldots, n-1\}$

と定義する．ここで，$CD_i(y_i, x_i)$の定義における“閉路に出会う路”とは，例えば，図2.1のグラフGの路$a^0a^5a^6a^5a^6a^7$のように閉路$a^5a^6a^5$を含む路，または，単純路$a^0a^5a^6a^7$のように閉路$a^5a^6a^5$（または閉路$a^7a^8a^9a^{10}a^7$）に接している路を意味する．A_0ファクトの引数の集合を

$$\overline{A_0} \;=\; \{(y_1, y_2) \mid A_0(y_1, y_2) \text{はファクト}\}$$

と記し，D_i, CD_i, F_iを使って，

$$ANS = \{sg(a_1, x_2) \mid \exists (y_1, y_2) \in \overline{A_0},\ D_1(y_1, a_1) \cap D_2(y_2, x_2) \neq \phi\},$$

$$CANS = \{sg(a_1, x_2) \mid\ \exists (y_1, y_2) \in \overline{A_0},\ CD_1(y_1, a_1) \cap CD_2(y_2, x_2) \neq \phi\},$$

$$FANS = \{sg(a_1, x_2) \mid \exists (y_1, y_2) \in \overline{A_0},\ F_1(y_1, a_1) \cap F_2(y_2, x_2) \neq \phi\}$$

と定義する．定義から分かるように，ANSは同世代問題の解集合である．CD_iは閉路を含む全ての路の長さを含むので，$CANS$は，G_1上の閉路を含む路とG_2上の閉路を含む路により生成される全ての解を包含する．$FANS$は長さ$(n-1)$以下の路により生成される解の集合であり，G_1，G_2の路の少なくとも一方が閉路を含まないような全ての解を包含する．従って，解集合ANSは$CANS$と$FANS$の和として計算することができる．

$$ANS = CANS \cup FANS.$$

$FANS$の計算は，有限集合F_iを扱えばよいので容易である．**計数法**（counting method）[BMSU86]により$O(ne)$時間で計算することができる[SPS87]．一方，$CANS$の計算は，無限集合CD_iを処理しなければならず問題がある．そこで，HaNa法 [HN88, HN91] では，$CD_i(y_i, x_i)$より計算の容易な**簡略化距離集合**（cyclic simplified distance set）$CS_i(y_i, x_i)$を利用して$CANS$を計算する方法を提案している．以下に説明する．

ある非負整数cおよび自然数の有限集合Nに対し，集合CDが

$$CD \;=\; \{c + \sum_{p \in N} \lambda_p p \mid \lambda_p : \text{非負整数}\}$$

と表されるとき，集合CDは**線形**（linear）であると言う．グラフG上の距離集合$CD(y, x)$は有限個の線形集合の和として表すことができる[HN88, HN91]．例えば，図2.1のグラフGには，単純路$a^0a^5a^6a^7$に単純閉路$a^5a^6a^5$と単純閉路$a^7a^8a^9a^{10}a^7$を任意個加えてできるa^0からa^7への路が存在するので，$CD(a^0, a^7)$は，集合$\{3 + 2\lambda_2 + 4\lambda_4 \mid \lambda_2, \lambda_4 : \text{非負整数}\}$を含む．同様に，単純路$a^0a^1a^2a^7$，$a^0a^1a^2a^3a^{10}a^7$とそれらの各々に出会う閉路を考えることにより，$CD(a^0, a^7)$は

$$\begin{aligned} CD(a^0, a^7) \;=\; & \{3 + 2\lambda_2 + 4\lambda_4 \mid \lambda_2, \lambda_4 : \text{非負整数}\} \\ & \cup\ \{3 + 4\lambda_4 \mid \lambda_4 : \text{非負整数}\} \cup \{5 + 4\lambda_4 \mid \lambda_4 : \text{非負整数}\} \end{aligned}$$

と表すことができる．

c, p'を非負整数，Nを自然数の有限集合とする．記号$L(c; p')$および$L(c; N)$を

$$L(c; p') \;=\; \{c + \lambda p' \mid \lambda : \text{非負整数}\}$$

$$L(c; N) \;=\; \{c + \sum_{p \in N} \lambda_p p \mid \lambda_p : \text{非負整数}\}$$

と定義する．（文献 [HN88. HN91] では，C を非負整数の有限集合とし，$L(C;p')$ および $L(C;N)$ が

$$\begin{aligned} L(C;p') &= \{c+\lambda p' \mid c\in C, \lambda : \text{非負整数}\} \\ L(C;N) &= \{c+\sum_{p\in N}\lambda_p p \mid c\in C, \lambda_p : \text{非負整数}\} \end{aligned}$$

として定義されている．本論文では，例えば，$L(C;p')$ は

$$L(C;p') = \bigcup_{c\in C} L(c;p')$$

として扱うことができるので，説明を簡単にするため，L の第1引数は集合 C ではなく非負整数 c を表すものとする．）また，N の全ての要素の最大公約数 g を $g=\gcd(N)$ と記す．$N_i, i=1,\ldots,d,$ を自然数のある有限集合とすると，先に述べたように，集合 $CD(y,x)$ は

$$CD(y,x) = \bigcup_{i=1}^{d} L(c_i;N_i) \tag{2.15}$$

と表されるので，これより，新しい集合

$$B(CD(y,x)) = B(\bigcup_{i=1}^{d} L(c_i;N_i)) = \bigcup_{i=1}^{d} L(c_i \bmod g_i; g_i) \tag{2.16}$$

を定義する．但し，$g_i=\gcd(N_i), i=1,\ldots,d,$ である．この集合 $B(CD(y,x))$ を $CD(y,x)$ の**簡略化距離集合**（cyclic simplified distance set）と呼び，$CS(y,x)$ と記す．上記の例では，

$$\begin{aligned} CS(a^0,a^7) &= B(CD(a^0,a^7)) \\ &= L(1;2)\cup L(3;4)\cup L(1;4) \\ &= \{1+2\lambda \mid \lambda : \text{非負整数}\}\cup\{3+4\lambda \mid \lambda : \text{非負整数}\} \\ &\quad \cup \{1+4\lambda \mid \lambda : \text{非負整数}\} \end{aligned} \tag{2.17}$$

である．

$CD(y,x)$ および $CS(y,x)$ の定義より，

$$CD(y,x)\subseteq CS(y,x).$$

$$|CS(y,x)-CD(y,x)|<\infty$$

を示すことができる [HN88, HN91]．従って，

$$CD_1(y_1,x_1)\cap CD_2(y_2,x_2)\neq o$$

$$\Leftrightarrow \quad |CD_1(y_1,x_1)\cap CD_2(y_2,x_2)|=\infty$$

$$\Leftrightarrow \quad |CS_1(y_1,x_1)\cap CS_2(y_2,x_2)|=\infty$$

$(\Leftarrow: |CS_i(y_i,x_i)-CD_i(y_i,x_i)|<\infty$ より．$\Rightarrow: CD_i(y_i,x_i)\subseteq CS_i(y_i,x_i)$ より）

$$\Leftrightarrow \quad CS_1(y_1,x_1)\cap CS_2(y_2,x_2)\neq o$$

が成立するので，$CANS$ を，簡略化距離集合 CS_i を使って，

$$CANS = \{sg(a_1,x_2) \mid \exists(y_1,y_2)\in\overline{A_0},\ CS_1(y_1,a_1)\cap CS_2(y_2,x_2)\neq o\}$$

として計算することができる．

2.3.2 簡略化距離集合の性質

簡略化距離集合 $CS(y,x)(=\bigcup_{i=1}^{d} L(c_i;p_i))$ は，具体的には，それを表す集合 $L(c_i,p_i)$ の組を計算することにより求められる．同じ $CS(y,x)$ を表す $L(c_i,p_i)$ の組は複数存在し得るが，HaNa 法はそのうちの一つを効率的に計算する．

HaNa 法が計算する $L(c_i,p_i)$ の組とその性質を説明する（この性質は拡張 HaNa 法の効率化に利用される）．

HaNa 法が計算する簡略化距離集合とその性質

ケース 1) 節点 x が G のある閉路に含まれる場合

式 (2.15), (2.16) の記号を用いる．HaNa 法は，$g_1,\ldots,g_d$ の最小公倍数 $p(=lcm(g_1,\ldots,g_d))$ および

$$C=\{c \bmod p \mid c\in\bigcup_{i=1}^{d} L(c_i \bmod g_i;g_i)\}$$

を計算し，そして，$CS(y,x)$ を

$$CS(y,x)=\bigcup_{c\in C} L(c;p)$$

と表す．$CS(y,x)$ のこの表現 $\bigcup_{c\in C} L(c;p)$ を $M(CS(y,x))$ と記す．例えば，式 (2.17) は，この表現により

$$CS(a^0,a^7)=L(1;4)\cup L(3;4)$$

と簡略化される．

（文献 [HN88, HN91] では，Gの**極大強連結成分**（strongly connected component）$\tilde{G}$に含まれる全ての閉路の長さの最大公約数を$\tilde{G}$の周期と定義しているが，）本論文では，この$p(=lcm(g_1,\ldots,g_d))$を$CS(y,x)$の**周期**（period）と呼ぶ．pは以下の性質を持つ．

(i) xを含むGの極大強連結成分を$\tilde{G}=(\tilde{V},\tilde{E})$（$\tilde{V}$：節点集合,$\tilde{E}$：枝集合）とする．節点$x'$が$\tilde{G}$に含まれるならば，($CD(y,x)=\bigcup_i L(c_i;N_i)$の$N_i$と$CD(y,x')=\bigcup_i L(c'_i;N_i)$の$N_i$は全て等しいので,）$CS(y,x)$の周期$p$と$CS(y,x')$の周期$p'$は等しい．

(ii) $\tilde{G}$に含まれる全ての単純閉路の長さを$\dot{p}_1,\ldots,\dot{p}_f$とする．$CD(y,x)=\bigcup_{i=1}^{d} L(c_i;N_i)$において，$\dot{p}_1,\ldots,\dot{p}_f$は全ての$N_i, i=1,\ldots,d$,に含まれるので,

$$p=lcm(\gcd(N_1),\ldots,\gcd(N_d))\leq \gcd(\{\dot{p}_1,\ldots,\dot{p}_f\})\leq|\tilde{V}| \tag{2.18}$$

が成立する．

ケース2) 節点xが閉路に含まれない場合

節点yからxへのある路$y\cdots z_i v_{i_1} v_{i_2}\cdots v_{i_c}x$において，節点$z_i$はある閉路に含まれ，節点$v_{i_1},v_{i_2},\ldots,v_{i_c}$は全てどの閉路にも含まれないとする．この条件を満足する全てのz_iを$z_1,\ldots,z_e$とし，各z_iの簡略化距離集合を（ケース1の節点であるので）

$$CS(y,z_i)=\bigcup_{c\in C_i} L(c;p_i) \qquad i=1,\ldots,e$$

と記す．HaNa法は

$$C'_i=\{(c+\text{路}\, z_i v_{i_1}\cdots v_{i_c}x\text{の長さ}) \bmod p_i \mid c\in C_i\}$$

を計算し，そして，$CS(y,x)$を

$$CS(y,x)=\bigcup_{i=1}^{e}\bigcup_{c\in C'_i} L(c;p_i)$$

と表す．

例えば，図2.1のGにおける$CS(a^0,a^{12})$を考えると，$y=a^0, x=a^{12}$に対し，$z_1=a^6, z_2=a^8$となり，$CS(a^0,a^6)=L(0;2)$，$CS(a^0,a^8)=L(0;4)\cup L(2;4)$であるので，HaNa法は$CS(a^0,a^{12})$を

$$CS(a^0,a^{12})=L(1;2)\cup L(2;4)\cup L(0;4)$$

と表す．□

図2.1のGにおいて，$y=a^0$から全てのxに対する$CS(a^0,x)$を求めると，

$$CS(a^0,a^0)=o.$$
$$CS(a^0,a^1)=L(1;4),$$
$$CS(a^0,a^2)=L(2;4),$$
$$CS(a^0,a^3)=L(3;4),$$
$$CS(a^0,a^4)=L(0;4),$$
$$CS(a^0,a^5)=L(1;2),$$
$$CS(a^0,a^6)=L(0;2),$$
$$CS(a^0,a^7)=L(1;4)\cup L(3;4),$$
$$CS(a^0,a^8)=L(0;4)\cup L(2;4),$$
$$CS(a^0,a^9)=L(1;4)\cup L(3;4),$$
$$CS(a^0,a^{10})=L(0;4)\cup L(2;4),$$
$$CS(a^0,a^{11})=L(1;4)\cup L(3;4).$$
$$CS(a^0,a^{12})=L(1;2)\cup L(0;4)\cup L(2;4)$$

となる．HaNa法は，グラフG上の一つの節点yとyから到達可能な全ての節点xとの間の簡略化距離集合$CS(y,x)$を最悪時間量$O(ne)$で計算する [HN88, HN91]．

2.4 拡張HaNa法

本節では，一般の$m(\geq 2)$の場合の同世代問題（第2.1節）の解法として，拡張HaNa法を提案する．HaNa法は第2.1節で述べたやり方により一般のmの問題に素直に拡張することができたが，しかし，そのように拡張して得られたHaNa法はマジック集合法より非効率であった．そこで，第2.4.1節では，第2.1節とは異なるやり方でHaNa法を一般のmに素直に拡張する．次に，その方法の一般のmには適用できない部分，および，非効率である部分を，各々，第2.4.2節，第2.4.3節で変更し，その結果得られるアルゴリズム（拡張HaNa法）を第2.4.4節で与える．拡張HaNa法の適用例は第2.4.5節に示す．

2.4.1 HaNa法との関係

$D_i(y_i, x_i), CD_i(y_i, x_i), F_i(y_i, x_i), CS_i(y_i, x_i)(i = 1, \ldots, m)$ を第2.3節と同様に定義する．
また，A_0 ファクトの引数の集合を

$$\overline{A_0} = \{(y_1, \ldots, y_m) \mid A_0(y_1, \ldots, y_m) \text{はファクト}\}$$

と記し，

$$\begin{aligned}
ANS &= \{sg(a_1, \ldots, a_h, x_{h+1}, \ldots, x_m) \mid \exists(y_1, \ldots, y_m) \in \overline{A_0} \\
&\quad (\bigcap_{i=1}^{h} D_i(y_i, a_i)) \cap (\bigcap_{i=h+1}^{m} D_i(y_i, x_i)) \neq \phi\}, \\
CANS &= \{sg(a_1, \ldots, a_h, x_{h+1}, \ldots, x_m) \mid \exists(y_1, \ldots, y_m) \in \overline{A_0} \\
&\quad (\bigcap_{i=1}^{h} CD_i(y_i, a_i)) \cap (\bigcap_{i=h+1}^{m} CD_i(y_i, x_i)) \neq \phi\}, \\
FANS &= \{sg(a_1, \ldots, a_h, x_{h+1}, \ldots, x_m) \mid \exists(y_1, \ldots, y_m) \in \overline{A_0} \\
&\quad (\bigcap_{i=1}^{h} F_i(y_i, a_i)) \cap (\bigcap_{i=h+1}^{m} F_i(y_i, x_i)) \neq \phi\}
\end{aligned}$$

と定義する．ANS は同世代問題の解集合であり，第2.3節と同じ議論により，

$$\begin{aligned}
ANS &= CANS \cup FANS, \\
CANS &= \{sg(a_1, \ldots, a_h, x_{h+1}, \ldots, x_m) \mid \exists(y_1, \ldots, y_m) \in \overline{A_0} \\
&\quad (\bigcap_{i=1}^{h} CS_i(y_i, a_i)) \cap (\bigcap_{i=h+1}^{m} CS_i(y_i, x_i)) \neq \phi\}
\end{aligned}$$

が成立する．

この結果にしたがって，第2.1節とは異なるやり方で一般の m に素直に拡張したHaNa法を図2.2に示す．図では，$V_i, i = 1, \ldots, m,$ をグラフ G_i の節点集合として，

$$\prod_{i=h+1}^{m} V_i = \{(x_{h+1}, \ldots, x_m) \mid x_{h+1} \in V_{h+1}, \ldots, x_m \in V_m\},$$

$$\prod_{i=1}^{h} CS_i(y_i, a_i) \times \prod_{i=h+1}^{m} CS_i(y_i, x_i) =$$

$$\{(L(c_1; p_1), \ldots, L(c_m; p_m)) \mid L(c_i; p_i) \subseteq CS_i(y_i, a_i), i = 1, \ldots, h,$$

$$L(c_i; p_i) \subseteq CS_i(y_i, x_i), i = h+1, \ldots, m\}$$

の記法を用いている．

step1: $FANS$ を計算する;
step2: /* $CANS$ の計算 */
　$CANS = \phi$;
　for$(\forall(y_1, \ldots, y_m) \in \overline{A_0})$ {　　/* ループ1 */
　step2.1: HaNa法のアルゴリズムにより，簡略化距離集合
　　$CS_i(y_i, x_i)(i = 1, \ldots, m, x_i \in V_i)$ を計算する;
　step2.2: /* $(y_1, \ldots, y_m)$ に対する $CANS$ の計算 */
　　for $(\forall(x_{h+1}, \ldots, x_m) \in \prod_{i=h+1}^{m} V_i)$ {　　/* ループ2 */
　　　if $(sg(a_1, \ldots, a_h, x_{h+1}, \ldots, x_m) \notin CANS)$ {
　　　　for $(\forall(L(p_1; q_1), \cdots L(p_m; q_m)) \in \prod_{i=1}^{h} CS_i(y_i, a_i) \times \prod_{i=h+1}^{m} CS_i(y_i, x_i))$ {
　　　　　/* ループ3 */
　　　　　if $(L(p_1; q_1) \cap \ldots \cap L(p_m; q_m) \neq \phi)$ {
　　　　　　$CANS = CANS \cup \{sg(a_1, \ldots, a_h, x_{h+1}, \ldots, x_m)\}$;
　　　　　　goto exitloop;
　　　　　}
　　　　}
　　　}
　　}
　　exitloop: ;　　/* ループ3を抜けるための空文 */
　　}
　}
step3: $ANS = FANS \cup CANS$;

図2.2: 一般の m に素直に拡張されたHaNa法

しかし，この図の方法には以下のような問題点がある．

(1) step1の$FANS$の計算を，$m=2$の場合，HaNa法は計数法を使って最悪時間量$O(ne)$で行う．一般のmの場合に計数法を適用するには，第2.1節で述べたやり方で拡張すればよいが，しかし，そのとき，計数法の最悪時間量は$O(n(e^h+e^{m-h}+t_0))$となる（少なくとも一つのG_iが閉路を含まないので，$FANS$の計算に必要な，積グラフ$G_1\times\cdots\times G_h$および$G_{h+1}\times\cdots\times G_m$の路の最大長が$n$以下になるからである）．$n\le e\le n^2$であるので，$h$が小さく，$e$が大きい場合，計数法の最悪時間量$O(n(e^h+e^{m-h}+t_0))$ $(\ge O(ne^{m-h}))$はマジック集合法の最悪時間量$O(e^hn^{m-h})$よりかなり大きくなり，従って，$FANS$を計算するアルゴリズムの改善が望まれる．

(2) step2のif文の

L式判定テスト: $L(c_1;p_1)\cap\ldots\cap L(c_m;p_m)\neq\phi$ (2.19)

を，$m=2$の場合，HaNa法は

$$L(c_1;p_1)\cap L(c_2;p_2)\neq\phi \;\Leftrightarrow\; \exists K(\text{整数})(c_1-c_2)/\gcd(p_1,p_2)=K \tag{2.20}$$

により行うことができるが [HH89]，この式は一般のmには適用できない．

(3) step2の主な計算時間はL式判定テストを行う時間である．L式判定テストは3重のforループ，すなわち，ループ1，ループ2，ループ3により繰り返し実行される．ループ1の繰り返し回数はt_0，ループ2の繰り返し回数はn^{m-h}である．また，$CS_i(y_i,x_i)$に含まれる異なる$L(c;p)$の数を$\|\,CS_i(y_i,x_i)\,\|$と記すと，(第2.3.2節の式(2.18)から示すことができるように)

$$\|\,CS_i(y_i,x_i)\,\|\le n$$

であるので [HN88, HN91]，ループ3の繰り返し回数は

$$\prod_{i=1}^{h}\|\,CS_i(y_i,a_i)\,\|\times\prod_{i=h+1}^{m}\|\,CS_i(y_i,x_i)\,\| \;\le\; n^m$$

である．従って，全てのL式判定テストを行う最悪時間量は$O(t_0n^{2m-h}T_{test})$となる．ここで，T_{test}は1回のL式判定テストを行う時間量である．たとえ，1回のL式判定テストを効率的に実行できた（すなわち，T_{test}を小さくできた）としても，hが小さいとき，L式判定テストの全回数$O(t_0n^{2m-h})$はマジック集合法の最悪時間量$O(e^hn^{m-h})$より大きいので，L

step1: ユークリッド互除法により$\gcd(p,p')$を計算する．よく知られているように，$\gcd(p,p')$はある整数δ,γを用いて

$$\gcd(p,p')=\delta p+\gamma p' \tag{2.21}$$

と表すことができ，このδもいっしょに計算する．

step2: $L(c;p)\cap L(c';p')\neq\phi$および式(2.20)より

$$\exists K(\text{整数})\quad (c-c')/\gcd(p,p')=K \tag{2.22}$$

が成立し，更に，この式(2.22)と式(2.21)より$\gcd(p,p')$を消去すると，

$$c-K\delta p=c'+K\gamma p'$$

が成立する．これより，$L(c'';p'')$は

$$\begin{aligned} p'' &= lcm(p,p')=pp'/\gcd(p,p'),\\ c'' &= (c-K\delta p)\bmod p'' \end{aligned}$$

と表すことができる．step1の$\gcd(p,p')$とδ，および，$K=(c-c')/\gcd(p,p')$を使って上記のp'',c''を計算する．

図2.3: $L(c'';p'')=L(c;p)\cap L(c';p')$を満たす$L(c'';p'')$の計算法

式判定テストの実行回数の減少が望まれる．

これらの問題点のうち，(1)に関しては，**逆計数法** (reverse counting method) [BMSU86]により，最悪時間量$O(t_0nme+t_0n|FANS|log|FANS|)$，最悪領域量$O(nm)$で$FANS$を計算することができる．(2)と(3)の問題点については，第2.4.2節と第2.4.3節で説明する．

2.4.2 L式判定テスト

初めに，$L(c;p)\cap L(c';p')\neq\phi$ $(0\le c<p, 0\le c'<p')$の場合に，

$$L(c'';p'')=L(c;p)\cap L(c';p')$$

を満たす$c''(\le p''),p''$を求める方法を図2.3に示す．この方法の時間量は，ユークリッド互除法にかかる$O(\log\max\{p,p'\})$ [Iba89, Knu81]である．

step1: /* 初期化 */
c_i, p_i を各々 c_i^1, p_i^1 と記す $(i = 1, \ldots, m)$.
$k := 1$;

step2: $k = \log m + 1$ (すなわち，$L(c_1; p_1) \cap \ldots \cap L(c_m; p_m) \neq \phi$ を計算済み) ならば，式 (2.19) は成立として終了.

step3: 式 (2.20) を使って，$L(c_{2i-1}^k; p_{2i-1}^k) \cap L(c_{2i}^k; p_{2i}^k) \neq \phi$ の判定を行う $(i = 1, \ldots, m/2^k)$. 少なくとも一つの i に対し $L(c_{2i-1}^k; p_{2i-1}^k) \cap L(c_{2i}^k; p_{2i}^k) = \phi$ であれば，式 (2.19) は不成立として終了.

step4: $L(c_i^{k+1}; p_i^{k+1}) = L(c_{2i-1}^k; p_{2i-1}^k) \cap L(c_{2i}^k; p_{2i}^k)$ を満足する c_i^{k+1}, p_i^{k+1} を図 2.3 の方法により求める $(i = 1, \ldots, m/2^k)$. $k = k + 1$; として step2 へ.

図 2.4: L 式判定テストを行うアルゴリズム

次に，図 2.3 の方法を使って，1 回分の L 式判定テスト (式 (2.19)) を行う方法を図 2.4 に示す．図 2.4 の方法の時間量は，step3，step4 におけるユークリッド互除法の時間量 $\log(\max\{p_{2i-1}^k, p_{2i}^k\})$ [Iba89, Knu81] の和である．$p_i^k \leq n^{2^{k-1}}$ であるので，

$$\sum_{k=1}^{\log m} \sum_{i=1}^{m/2^k} O(\log(\max\{p_{2i-1}^k, p_{2i}^k\})) \;=\; \sum_{k=1}^{\log m} \sum_{i=1}^{m/2^k} O(\log n^{2^{k-1}}) \;=\; O(m \log m \log n)$$

となる.

更に，図 2.4 の方法の使い方を工夫することにより，1 回の L 式判定テストの時間量を $O(m \log m \log n)$ から $O(m \log n)$ に減らせることを示す．各 $(y_1, \ldots, y_m)(\in \overline{A_0})$ に対する複数回の L 式判定テスト ...，$L(c_1; p_1) \cap \ldots \cap L(c_{m-1}; p_{m-1}) \cap L(c_m; p_m) \neq \phi$，$L(c_1; p_1) \cap \ldots \cap L(c_{m-1}; p_{m-1}) \cap L(c'_m; p'_m) \neq \phi$, ... において，前回の L 式判定テスト $L(c_1; p_1) \cap \ldots \cap L(c_m; p_m) \neq \phi$ の中間データを記憶し，かつ，今回の L 式判定テスト $L(c_1; p_1) \cap \ldots \cap L(c'_m; p'_m) \neq \phi$ が前回の L 式判定テストと一つの $L(c; p)$ しか替わらないようにする (すなわち，$L(c'_m; p'_m)$ のみ $L(c_m; p_m)$ と異なるようにする) . このようにすると，図 2.4 の step3, step4 は一つの i に対してのみ計算すればよいので，従って，各 $(y_1, \ldots, y_m)(\in \overline{A_0})$ に対する 2 回目以降の L 式判定テストにおいて，1 回の時間量を $\sum_{k=1}^{\log m} O(\log n^{2^{k-1}}) = O(m \log n)$ に減らすことができる.

2.4.3 L 式判定テストの回数の減少

初めに補題を示す.

補題 2.1 グラフ $G = (V, E)$ において節点 y を一つ固定する．y と G の全ての節点との間の簡略化距離集合の和を

$$SS(y) \;=\; \bigcup_{x \in V} CS(y, x) = \bigcup_{i} L(c_i; p_i)$$

と記し，その中に含まれる異なる式 $L(c_i; p_i)$ の数を $\| SS(y) \|$ と記す．そのとき，

$$\| SS(y) \| \;\leq |V| (= n) \tag{2.23}$$

が成立する.

証明 (i) $L(c_i; p_i)$ に現れる異なる p_i を，一般性を失うことなく，$p_1, p_2, \ldots, p_{d''}$ とする．初めに，

$$p_1 + p_2 + \cdots + p_{d''} \leq n \tag{2.24}$$

を示す．G の閉路を含む極大強連結成分を $\tilde{G}^i = (\tilde{V}^i, \tilde{E}^i), i = 1, \ldots, f$, とすると，

$$SS(y) \;=\; \Big(\bigcup_{x \in \tilde{V}^1 \cup \cdots \cup \tilde{V}^f} CS(y, x) \Big) \cup \Big(\bigcup_{x \in V - (\tilde{V}^1 \cup \cdots \cup \tilde{V}^f)} CS(y, x) \Big)$$

と表せる．ここで，第 1 項 $\bigcup_{x \in \tilde{V}^1 \cup \cdots \cup \tilde{V}^f} CS(y, x)$ は閉路に含まれる全ての節点の簡略化距離集合の和であり，第 2 項 $\bigcup_{x \in V - (\tilde{V}^1 \cup \cdots \cup \tilde{V}^f)} CS(y, x)$ は閉路に含まれない全ての節点の簡略化距離集合の和である．第 2.3.2 節のケース 2 で紹介したように，第 2 項の中に現れる任意の $L(c'; p)$ の p は第 1 項の中のある $L(c; p)$ の p と等しい．また，同じく第 2.3.2 節のケース 1 で紹介したように，第 1 項の各 $\bigcup_{x \in \tilde{V}^i} CS(y, x)$ の中に現れる $L(c; p)$ の p は全て等しく (それを p'_i とおく)，$p'_i \leq |\tilde{V}^i|$ である．従って，

$$\begin{aligned} & p_1 + p_2 + \cdots + p_{d''} \\ & \leq p'_1 + p'_2 + \cdots + p'_f (i \neq j \text{に対し} p'_i = p'_j \text{の場合も有り得るから}) \\ & \leq |\tilde{V}^1| + \cdots + |\tilde{V}^f| \leq |V| = n \end{aligned}$$

である.

(ii) 次に，任意の $L(c;p)$ において，$0 \leq c \leq p-1$ であるので，

$$(\text{同じ} p_i \text{をもつ異なる} L(c_i;p_i) \text{の数}) \leq p_i \tag{2.25}$$

である．

式 (2.24) と (2.25) より式 (2.23) が証明された．□

補題を利用してL式判定テストの回数を減らすことができる．図2.2のstep2の3重のforループのうち，ループ2では，各 $(x_{h+1},\ldots,x_m)(\in \prod_{i=h+1}^{m} V_i)$ ごとにL式判定テスト（式(2.19)）を行う．このため，同じL式判定テストが異なる $(x_{h+1},\ldots,x_m)$ に対して重複して行われる場合がある．拡張HaNa法は，この重複を防ぐために，各 $(y_1,\ldots,y_m)(\in \overline{A_0})$ に対し $SS_i(y_i), i=h+1,\ldots,m,$ を作り，$SS_i(y_i)$ の中の式 $L(c_i,p_i)$ の重複を取り除いた後，L式判定テストを行う．同じ $L(c_1,p_1),\ldots L(c_m,p_m)$ の組に対する重複テストを除くようにすれば，補題より，L式判定テストの全回数を，

$$|\overline{A_0}| \times \prod_{i=1}^{h} \| CS_i(y_i,a_i) \| \times \prod_{i=h+1}^{m} \| SS_i(y_i) \| = O(t_0 n^m)$$

回に減らすことができる．計算時間は $O(t_0 n^m T_{test})$ である．

2.4.4 拡張HaNa法

以上の議論をまとめ，拡張HaNa法を図2.5に与える．但し，

$$NN_i(L(c_i;p_i),y_i) = \{x_i \mid L(c_i;p_i) \subseteq CS_i(y_i,x_i)\}, \qquad i=h+1,\ldots,m,$$

$$\prod_{i=h+1}^{m} NN_i(L(c_i;p_i),y_i) =$$
$$\{(x_{h+1},\ldots,x_m) \mid x_i \in NN_i(L(c_i;p_i),y_i), i=h+1,\ldots,m\},$$

$$\prod_{i=h+1}^{m} SS_i(y_i) =$$
$$\{(L(c_{h+1};p_{h+1}),\ldots,L(c_m;p_m)) \mid L(c_i;p_i) \subseteq SS_i(y_i), i=h+1\ldots,m\}$$

とする．

2.4.5 適用例

$m=3, h=2$，問い合わせアトムを $sg(a^5,a^7,X)?$ とする．また，$\overline{A_0}=\{(a^0,a^0,a^0)\}$ とし，$G_i, i=1,2,3,$ は全て図2.1の G に等しいとする．

step1: 逆計数法を使って $FANS$ を計算する;
step2: /* $CANS$ の計算 */
  $CANS=\phi$;
  for$(\forall(y_1,\ldots,y_m)\in\overline{A_0})$ {　　/* ループ1 */
  step2.1: HaNa法のアルゴリズムにより，簡略化距離集合
    $CS_i(y_i,x_i)(i=1,\ldots,m,x_i\in V_i)$ を計算する;
  step2.2: 各グラフ $G_i, i=h+1,\ldots,m,$ において，$SS_i(y_i)$ および
    $NN_i(L(c_i;p_i),y_i)(L(c_i;p_i)\subseteq SS_i(y_i))$ を計算する;
  step2.3: /* $(y_1,\ldots,y_m)$ に対する $CANS$ の計算 */
    for $(\forall(L(c_{h+1};p_{h+1}),\ldots,L(c_m;p_m))\in\prod_{i=h+1}^{m} SS_i(y_i))$ {　/* ループ2 */
      for $(\forall(L(c_1;p_1),\ldots,L(c_h;p_h))\in\prod_{i=1}^{h} CS_i(y_i,a_i))$ {　/* ループ3 */
        if $(L(c_1;p_1)\cap\cdots\cap L(c_m;p_m)\neq\phi)$ {
          $CANS=CANS\cup\{sg(a_1,\ldots,a_h,x_{h+1},\ldots,x_m)\mid(x_{h+1},\ldots,x_m)$
            $\in\prod_{i=h+1}^{m} NN_i(L(c_i;p_i),y_i)\}$;
          goto exitloop;
        }
      }
    exitloop: ; /* ループ3を抜けるための空文 */
    }
  }
step3: $ANS=FANS\cup CANS$;

図2.5: 拡張HaNa法

初めに$CANS$を求める（図2.5のstep2）．$\overline{A_0}=\{(a^0,a^0,a^0)\}$であるので，ループ1すなわちstep2.1からstep2.3は，$(y_1,y_2,y_3)=(a^0,a^0,a^0)$に対し1回だけ実行される．step2.1で計算される簡略化距離集合$CS_1(a^0,a^j),CS_2(a^0,a^j),CS_3(a^0,a^j),j=0,\ldots,12$，は全て，第2.3.2節で求めた$CS(a^0,a^j)$に等しい．従って，step2.2の$SS_3(a^0)$は

$$\begin{aligned} SS_3(a^0) &= \bigcup_{j=0}^{12} CS_3(a^0,a^j) \\ &= L(0;2)\cup L(1;2)\cup L(0;4)\cup L(1;4)\cup L(2;4)\cup L(3;4) \end{aligned}$$

である．また，NN_3については，例えば，$L(1;2)$は$CS(a^0,a^5)$と$CS(a^0,a^{12})$に含まれるので，

$$NN_3(L(1;2),a^0) = \{a^5,a^{12}\}$$

であり，その他のNN_iも同様に計算すると，

$$\begin{aligned} NN_3(L(0;2),a^0) &= \{a^6\}, \\ NN_3(L(0;4),a^0) &= \{a^4,a^8,a^{10},a^{12}\}, \\ NN_3(L(1;4),a^0) &= \{a^1,a^7,a^9,a^{11}\}, \\ NN_3(L(2;4),a^0) &= \{a^2,a^8,a^{10},a^{12}\}, \\ NN_3(L(3;4),a^0) &= \{a^3,a^7,a^9,a^{11}\} \end{aligned}$$

である．step2.3のループ2の$SS_3(y_3)$は$SS_3(a^0)$，また，ループ3の$CS_1(y_1,a_1)$と$CS_2(y_2,a_2)$は，各々，$CS_1(a^0,a^5)$，$CS_2(a^0,a^7)$である．従って，例えば，$L(1;2)(\in CS_1(a^0,a^5))$，$L(1;4)(\in CS_2(a^0,a^7))$，$L(1;2)(\in SS_3(a^0))$に対し，L式判定条件:

$$L(1;2)\cap L(1;4)\cap L(1;2) \neq \phi$$

が成立する．同様に，$L(1;4),L(3;4)$ $(\in SS_3(a^0))$ に対しても，L式判定条件が成立し，その結果，

$$\begin{aligned} CANS &= \{sg(a^5,a^7,x) \mid \\ &\qquad x\in NN_3(L(1;2),a^0)\cup NN_3(L(1;4),a^0)\cup NN_3(L(3;4),a^0)\} \\ &= \{sg(a^5,a^7,x) \mid x\in\{a^1,a^3,a^5,a^7,a^9,a^{11},a^{12}\}\} \\ &= \{sg(a^5,a^7,a^1),sg(a^5,a^7,a^3),sg(a^5,a^7,a^5),sg(a^5,a^7,a^7), \\ &\qquad sg(a^5,a^7,a^9),sg(a^5,a^7,a^{11}),sg(a^5,a^7,a^{12})\} \end{aligned}$$

を得る．

本例では，$FANS$の計算結果は$CANS$と等しく，$ANS=CANS$となる．

2.5 最悪計算量の解析

(1) 図2.5のstep1: $FANS$を計算する逆計数法 [BMSU86] の最悪時間量は$O(t_0nmc+t_0n|FANS|\log|FANS|)$，最悪領域量は$O(nm)$である．

(2) 図2.5のstep2.1: iおよびy_iを固定したときの$CS_i(y_i,x_i)(\forall x_i\in V_i)$を計算する最悪時間量は$O(ne)$であるので [HN88, HN91]，step2.1の最悪時間量は$O(t_0mne)$，最悪領域量は$O(mne)$である．

(3) 図2.5のstep2.2: iおよびy_iを固定し，全ての$x_i(\in V_i)$について$CS_i(y_i,x_i)$を調べながら，一つの$SS_i(y_i)$を計算する時間は$\sum_{x_i\in V_i}\parallel CS_i(y_i,x_i)\parallel\log\parallel SS_i(y_i)\parallel$である．また，文献 [HN88, HN91] および第2.4.3節の補題より$\parallel CS_i(y_i,x_i)\parallel\leq\parallel SS_i(y_i)\parallel\leq n$である．従って，$t_0$個の$(y_1,\ldots,y_m)\in\overline{A_0}$に対して$SS_{h+1}(y_{h+1}),\ldots,SS_m(y_m)$を計算する時間$Time1$は，

$$Time1 = t_0(\sum_{i=h+1}^{m}\sum_{x_i\in V_i}\parallel CS_i(y_i,x_i)\parallel\log\parallel SS_i(y_i)\parallel) = O(t_0(m-h)n^2\log n)$$

であり，また，そのとき必要となる領域量$Space1$は

$$Space1=\sum_{i=h+1}^{m}\parallel SS_i(y_i)\parallel = O((m-h)n)$$

である．

各$NN_i(L(c_i;p_i),y_i)$の計算は，$CS_i(y_i,x_i)$より$SS_i(y_i)$を作る際に$L(c_i;p_i)$からx_iへのポインタをはることにより，計算時間を増やすことなく，$SS_i(y_i)$の計算と同時に進めることができる．そのとき必要となる領域量$Space2$は，$NN_i(L(c_i;p_i),y_i)\leq|V_i|=n$より，

$$Space2 = \sum_{i=h+1}^{m}\sum_{L(c_i;p_i)\subseteq SS_i}|NN_i(L(c_i;p_i),y_i)| = O((m-h)n^2)$$

である．

以上より，step2.2の最悪時間量は$O(t_0(m-h)n^2\log n)$，最悪領域量は$Space1$と$Space2$より$O((m-h)n^2)$である．

(4) 図2.5のstep2.3: 第2.4.2節より1回のL式判定テストに必要な時間量は$O(m\log n)$，また，第2.4.3節よりL式判定テストの全回数は$O(t_0n^m)$である．従って，全てのL式判定

テストに必要な時間$Time3$は，

$$Time3 = O(t_0 n^m m \log n)$$

である．

次に，式

$$CANS = CANS \cup \{sg(a_1, \ldots, a_h, x_{h+1}, \ldots, x_m) \mid (x_{h+1}, \ldots, x_m) \in \prod_{i=h+1}^{m} NN_i(L(c_i; p_i), y_i)\} \quad (2.26)$$

を計算するのに必要な時間量$Time4$について考える．式(2.26)は，ループ3の繰り返しでは高々1回しか実行されない．従って，式(2.26)の全実行回数は，$|\overline{A_0}| \prod_{i=h+1}^{m} \| SS_i(y_i) \|$である．また，1回の実行に必要な時間は，$\prod_{i=h+1}^{m} |NN_i(L(c_i; p_i), y_i)| (\leq |CANS|)$個の解$sg(a_1, \ldots, a_h, x_{h+1}, \ldots, x_m)$を集合$CANS$に重複を除きながら格納する時間であり，一つの解を$CANS$に格納する時間は，$CANS$の要素（すなわち解）を整列しておけば$O(\log |CANS|)$である．従って，

$$\begin{aligned} Time4 &= |\overline{A_0}| \prod_{i-h+1}^{m} \| SS_i(y_i) \| \prod_{i=h+1}^{m} |NN_i(L(c_i; p_i), y_i)| \log |CANS| \\ &= O(t_0 n^{m-h} |CANS| \log |CANS|) \end{aligned}$$

である．

$Time3$と$Time4$を合わせ，step2.3の最悪時間量は$O(t_0 n^m m \log n + t_0 n^{m-h} |CANS| \log |CANS|)$である．また，step2.3において必要となる領域量は$CANS$を記憶するためのものであり，$O(|CANS|)$である．

(5) 図2.5のstep3:　最悪時間量は$O(|ANS| \log |ANS|)$，最悪領域量は$O(|ANS|)$である．

(6) 以上，(1)から(5)をまとめると，拡張HaNa法の最悪時間量は

$$O(t_0(mne + n^m m \log n + n^{m-h} |ANS| \log |ANS|)),$$

最悪領域量は

$$O(mnc + |ANS|)$$

である．ここで，mを定数と見なしてmについての多項式項を消去すると，$m \geq 3$のとき最悪時間量は

$$O(t_0(n^m \log n + n^{m-h} |ANS| \log n))$$

になる．

2.6　最悪計算量の比較

$m \geq 3$，かつ，hとeが大きく，t_0と$|ANS|$が小さい場合，最悪時間量において，拡張HaNa法がマジック集合法より効率的であることを示す（$m = 2$の場合，拡張HaNa法はマジック集合法およびHaNa法と同程度，または，それより非効率である）．

(1) $m \geq 3$，かつ，hとeが大きく，t_0と$|ANS|$が小さい場合の最悪時間量

$m \geq 3$とする．拡張HaNa法の最悪時間量$O(t_0(n^m \log n + n^{m-h} |ANS| \log n))$とマジック集合法の最悪時間量$O(e^h n^{m-h})$のどちらが小さいかは，対象とする問題の性質（すなわち，$h, m, t_0, n, e, |ANS|$の値）に依存する．二つの条件$|ANS| \leq n^h$（条件1と呼ぶ）と$(t_0 \log n)^{1/h} \leq e/n$（条件2と呼ぶ）が成立する程度に，$h$と$e$が大きく，$t_0$と$|ANS|$が小さい場合を考える．条件1と2が成立するとき，拡張HaNa法の最悪時間量とマジック集合法のそれとの間には，

$$\begin{aligned} & O(t_0(n^m \log n + n^{m-h} |ANS| \log n)) \\ = \; & O(t_0 n^m \log n) \quad （条件1より） \\ \leq \; & O(e^h n^{m-h}) \quad （条件2より） \end{aligned}$$

が成立するので，最悪時間量において，拡張HaNa法はマジック集合法より優れる．

なお，よく議論される問題例の中には，以下に説明するように，$h, e, t_0, |ANS|$に関する上記の条件1と2が成立するような問題例が多くあることを注意しておく．よく議論される問題例では，第2.2節の問題例のように，sgの初期値を与えるA_0ファクト集合は$\overline{A_0} = \{(y, \ldots, y) \mid y$は任意の定数$\}$として与えられることが多く，このとき，$t_0 = n$である．更に，$n \leq e \leq n^2$であるので，$e$が$n$にあまり近くなく，かつ，$h$が大きい場合，$t_0 = n$のこのような問題例の中には，条件2（すなわち，$(n \log n)^{1/h} \leq e/n$）が成立するような問題例が多くある．また，解の数$|ANS|$は通常あまり大きくないので，$h$が大きい場合，条件2に加えて更に条件1も成立するような問題例は多くある．このように，よく議論される問題例の中には，$h, e, t_0, |ANS|$に関する上記の条件1と2が成立するような問題例は多くある．

(2) $m \geq 3$の場合の最悪領域量

いかなるアルゴリズムも結果を貯えるために$|ANS|$の領域量を必要とするので，比較的小さな領域量$O(mne)$を無視すれば，拡張HaNa法の領域量はほぼ最適である（$O(mne)$はマジック集合法の最悪領域量$O(n^m)$と比べ小さい）．

2.7 むすび

再帰述語の引数の数m（≥ 2）が一般の整数である場合の同世代問題を解くための方法として，$m=2$の場合に対して提案されているHaNa法を変形して拡張HaNa法を提案し，その最悪計算量を解析した．そして，従来の方法の中で最も効率的と思われるマジック集合法と拡張HaNa法を比較し，$m \geq 3$，かつ，通常の多くの問題例に見られるように，hとeが大きく，t_0と$|ANS|$が小さい場合，最悪時間量において，拡張HaNa法がマジック集合法より効率的であることを示した．ここで，hは問い合わせアトム中の定数の数，eは再帰ルールに現れる各EDB述語についてのファクトの数，t_0は初期値を与えるEDB述語についてのファクトの数，$|ANS|$は解の数である．また，拡張HaNa法の最悪領域量がほぼ最適であることも示した．

第3章

直積問題を効率的に解くための問い合わせ処理法

右線形問題クラスや同世代問題クラスを包含し，かつ，一部の非線形問題も含むdatalog問題クラスの新しい部分クラス（直積問題クラスと呼ぶ）を定義し，この問題クラスを効率的に解くための方法として直積法を提案する．

pを述語，$C_1,\ldots,C_h$の各々を定数組の集合，$C_1 \times \cdots \times C_h$を集合$C_1,\ldots,C_h$の直積とするとき，式$p[C_1 \times \cdots \times C_h]$は基礎アトム（gaと記す）の集合$\{p(\boldsymbol{c}) \mid \boldsymbol{c} \in C_1 \times \cdots \times C_h\}$を表すと定義し，この式をgaseと呼ぶ．

直積問題では，gaを生成する代わりにgaseを生成し，生成可能なすべてのgaの集合をそれらgaseの和として表すことができる．故に，それらgaseから解集合を求めることができる．gaseを生成する場合，一つのgaseで多数のgaを表すことができるので，生成するgaseの数は少なくなる可能性があり，故に，問題を効率的に解ける可能性がある．直積法は，この考えに基づいて，gaseを生成することによって，直積問題を効率的に解こうとする方法である．

直積問題クラス全体に適用できる従来の方法の中で，マジック集合法は最も効率的な方法と思われる．直積問題の例として，同世代問題の再帰ルールを複数にした問題，および，非線形にした問題を考え，これらの直積問題を使って，直積法とマジック集合法の効率を比較する．計算機実験により，どちらの問題においても，ファクト集合がファクトを密に含む問題例に対し，直積法がマジック集合法より効率的であることを示す．

3.1 はじめに

演繹データベースにおける問い合わせ処理の効率化を目的として，適用範囲は狭いが大変効率的な方法から，前者ほど効率的ではないが適用範囲の広い方法まで，様々な方法が提案されている [BR86, NK88, Ull88, Ull89, MS90]．例えば，前者の方法には，右線形問題クラスを主な対象とするNRSU法 [NRSU89b]，同世代問題クラスを対象とする計数法 [BMSU86]（第1.4.2節を参照）等があり，後者の方法には，一般の問題（すなわち，datalog問題クラスと一部のホーン問題）を対象とするマジック集合法 [BMSU86, BR87, Ram88]（第1.4.3節を参照）等がある．ここで，右線形問題クラスと同世代問題クラス（第1.4.2節を参照）はdatalog問題クラスの部分クラスであり，datalog問題クラスはホーン問題クラスの部分クラスである（第1.3節を参照）．例に挙げた方法も含め，提案された方法の多くは，生成する中間データの数を減らすことにより，問い合わせ処理の効率化を計っている．これらの方法が中間データの数を減らすために用いる手法は様々に異なっているが，しかし，いずれの方法も中間データとして**基礎アトム**（gaと記す）を生成する．

本章では，右線形問題クラスや同世代問題クラスを包含し，かつ，一部の非線形問題も含むdatalog問題クラスの新しい部分クラス（**直積問題**（Cartesian product problem）クラスと呼ぶ）を定義し，この問題クラスを効率的に解くための方法として**直積法**（Cartesian product method）を提案する．

pを述語，$C_1,\ldots,C_h$の各々を定数組の集合，$C_1\times\cdots\times C_h$を集合$C_1,\ldots,C_h$の**直積**（Cartesian product），すなわち，$C_1\times\cdots\times C_h=\{(c_{11},\ldots,c_{1m_1},\ldots,c_{h1},\ldots,c_{1m_h})\mid(c_{11},\ldots,c_{1m_1})\in C_1,\ldots,(c_{h1},\ldots,c_{1m_h})\in C_h\}$とするとき，式$p[C_1\times\cdots\times C_h]$はga集合$\{p(\boldsymbol{c})\mid\boldsymbol{c}\in C_1\times\cdots\times C_h\}$を表すと定義し，この式$p[C_1\times\cdots\times C_h]$を**直積型基礎アトム集合式**（ground atom set expression of Cartesian product type），略して，**gase**と呼ぶ．

意味IMPは問題から生成できるすべてのgaの集合であり，解集合ANSは問い合わせアトムによって表される条件を満足する，IMPの中のgaの集合である．直積問題では，gaの代わりにgase（すなわち，$p,C_1,\ldots,C_h$の組）を生成し，意味IMPをそれらgaseが表すga集合の和として表すことができる．故に，それらgaseの中から解集合ANSを求めることができる．

例えば，ルール集合が

$$s(X,Y)\leftarrow F(X,Y). \tag{3.1}$$

$$s(X,Y)\leftarrow A(X',X),B(Y',Y),s(X',Y'). \tag{3.2}$$

である問題は直積問題である．そのファクト集合を，例えば

$$\{F(a,b)\}\cup\{A(a,a_i)\mid i=1,\ldots,h_1\}\cup\{A(a_i,a_{ij})\mid i=1,\ldots,h_1,j=1,\ldots,h_2\}$$
$$\cup\{B(b,b_{i'})\mid i'=1,\ldots,k_1\}\cup\{B(b_{i'},b_{i'j'})\mid i'=1,\ldots,k_1,j'=1,\ldots,k_2\}$$

とする（簡単のため，定数$a,a_1,\ldots,a_{h_1},a_{11},\ldots,a_{h_1h_2},b,b_1,\ldots,b_{k_1},b_{11},\ldots,b_{k_1k_2}$はすべて異なると仮定する）と，次のようにgaseを生成することができる．(1) ルール(3.1)からgaの初期値$s(a,b)$が生成できるので，このga $s(a,b)$に対しgaseの初期値$s[\{a\}\times\{b\}]$を生成する．(2) 次に，このgase $s[\{a\}\times\{b\}]$をルール(3.2)の本体に代入して，次のように，頭部アトムのためのgaseを生成する．ルール(3.2)の本体アトム$s(X',Y')$の第1引数X'に集合$\{a\}$の中の任意の定数（この場合aのみ）を代入し，本体アトム$A(X',X)$に代入可能な任意のAファクトを代入して，頭部アトム$s(X,Y)$の第1引数Xの値の集合を計算すると，定数の集合$C_1'=\{a_i\mid i=1,\ldots,h_1\}$が得られる．同様に，本体アトム$s(X',Y')$の第2引数$Y'$に集合$\{b\}$の中の任意の定数（この場合$b$のみ）を代入し，本体アトム$B(Y',Y)$に代入可能な任意の$B$ファクトを代入して，頭部アトム$s(X,Y)$の第2引数$Y$の値の集合を計算すると，定数集合$C_2'=\{b_{i'}\mid i'=1,\ldots,k_1\}$が得られる．これらの$C_1'$と$C_2'$よりgase $s[C_1'\times C_2']$を生成する．(3) 再び，得られたgase $s[C_1'\times C_2']$をルール(3.2)の本体に代入して，(2)のときと同様のやり方で，頭部アトムのためのgaseを生成する．頭部アトム$s(X,Y)$の引数Xのための値の集合として$C_1''=\{a_{ij}\mid i=1,\ldots,h_1,j=1,\ldots,h_2\}$が得られ，引数$Y$のための値の集合として$C_2''=\{b_{i'j'}\mid i'=1,\ldots,k_1,j'=1,\ldots,k_2\}$が得られるので，gase $s[C_1''\times C_2'']$を生成する．(4) 更に，gase $s[C_1''\times C_2'']$をルール(3.2)の本体に代入して，(2)，(3)のときと同様のやり方でgaseを生成しようとすると，定数はすべて異なるという仮定より，頭部アトムの引数のための集合を生成することができず，gaseの生成に失敗する．以上，三つのgase $s[\{a\}\times\{b\}],s[C_1'\times C_2'],s[C_1''\times C_2'']$を生成することができる．これら三つのgaseが表すga集合の和$\{s(a,b)\}\cup\{s(a_i,b_{i'})\mid a_i\in C_1',b_{i'}\in C_2'\}\cup\{s(a_{ij},b_{i'j'})\mid a_{ij}\in C_1'',b_{i'j'}\in C_2''\}$はこの問題の意味に一致する．

gaseの定義から分かるように，gaseは一つのgaseで多数のgaを表すことができるので，中間データとしてgaseを生成することで，生成する中間データの数を小さくできる可能性がある．一つのgaseを生成し処理する時間は一つのgaのそれに比べ大きいが，もし，生成するgaseの数が大変小さいならば，問い合わせ処理の計算時間を全体として減らせる可能

性がある．直積法は，この考えに基づいて，中間データとしてgaseを生成することによって，直積問題を効率的に解こうとする方法である．

従来の方法は，先に述べたように，中間データとしてgaを生成する．一方，直積法は，上に述べたように，gaseを生成する．gaseを生成することが直積法の特徴である．

直積問題クラス全体に適用できる従来の方法には，マジック集合法等，一般の問題を対象とした方法がある．これらの方法の中で，マジック集合法は最も効率的な方法の一つと思われる．直積問題の例として，同世代問題の再帰ルールを複数にした問題，および，同世代問題の再帰ルールを非線形にした問題を考え，これらの直積問題を使って，直積法とマジック集合法の効率を比較する．ファクト集合の中のファクトの数がファクト集合に現れる異なる定数の数に比べて大きい場合，そのファクト集合は“ファクトを密に含む”と言う．計算機実験により，どちらの問題においても，ファクト集合がファクトを密に含む問題例に対し，直積法がマジック集合法より効率的であることを示す．なお，どちらの問題においても，最悪時間量においては，直積法はマジック集合法に劣る．

以後，特に断らない限り，定数を$a, a', a_1, \ldots, c, c', c_1, \ldots$，定数組の集合を$C, C', C_1, \ldots$，定数組の集合の直積（例えば$C_1 \times \cdots \times C_h$）を$\boldsymbol{C}, \boldsymbol{C}', \boldsymbol{C}_1, \ldots$と記す．

本章の以降は次のように構成される．第3.2節で直積問題クラスを定義し，第3.3節で直積法を与え，第3.4節で直積法の適用例を示す．第3.5節で実験を説明し，第3.6節でまとめる．

3.2 直積問題

準備として，用語を幾つか定義した後，直積問題を定義する．その後，直積問題の例を与え，最後に，直積問題クラスの広さについて説明する．

述語（$X_1, \ldots, X_m$を異なる変数として$p(X_1, \ldots, X_m)$と記す）に対し，その引数$X_1, \ldots, X_m$を複数のグループ（$\{X_{11}, \ldots, X_{1m_1}\}, \ldots, \{X_{h1}, \ldots, X_{hm_h}\}$と記す）に分割することにより，引数$X_{11}, \ldots, X_{1m_1}$をもつ述語（$p_1(X_{11}, \ldots, X_{1m_1})$と記す）と，…，引数$X_{h1}, \ldots, X_{hm_h}$をもつ述語（$p_h(X_{h1}, \ldots, X_{hm_h})$と記す）を作ることを，**述語を分割する**（decompose a predicate）と言い，式$p(X_1, \ldots, X_m) \to p_1(X_{11}, \ldots, X_{1m_1}) \wedge \cdots \wedge p_h(X_{h1}, \ldots, X_{hm_h})$によって示す．ここで，$\{X_1, \ldots, X_m\} = \{X_{11}, \ldots, X_{1m_1}\} \cup \cdots \cup \{X_{h1}, \ldots, X_{hm_h}\}$であり，かつ，変数$X_{11}, \ldots, X_{1m_1}, \ldots, X_{h1}, \ldots, X_{hm_h}$はすべて異なる．

アトム$p(X'_1, \ldots, X'_m)$（ここで，$X'_1, \ldots, X'_m$には同じ変数が含まれてよい）とその述語pについての述語分割$p(X_1, \ldots, X_m) \to p_1(X_{11}, \ldots, X_{1m_1}) \wedge \cdots \wedge p_h(X_{h1}, \ldots, X_{hm_h})$が与えられている．このとき，述語分割の左辺のアトム$p(X_1, \ldots, X_m)$にアトム$p(X'_1, \ldots, X'_m)$を代入することにより，述語分割の右辺のアトムとして，アトム$p_1(X'_{11}, \ldots, X'_{1m_1})$と，…，アトム$p_h(X'_{h1}, \ldots, X'_{hm_h})$を作ることを，**アトムを述語分割にしたがって分割する**と言う．例えば，アトム$p(X, Y, Y, Z)$を述語分割$p(X_1, X_2, X_3, X_4) \to p_1(X_1, X_2) \wedge p_2(X_3, X_4)$にしたがって分割すると，アトム$p_1(X, Y)$とアトム$p_2(Y, Z)$が作られる．

問題に現れる各IDB述語について，その述語分割が与えられているとし，それら述語分割の集合をPDと記す．ここで，少なくとも一つの述語については，述語を実質的に分割するものとする．すなわち，$h \geq 2$である述語分割$p(X_1, \ldots, X_m) \to p_1(X_{11}, \ldots, X_{1m_1}) \wedge \cdots \wedge p_h(X_{h1}, \ldots, X_{hm_h})$が与えられているとする．また，述語を実質的に分割しないような述語があってもよいが，そのような述語については，簡単のため，述語分割$p(X_1, \ldots, X_m) \to p(X_1, \ldots, X_m)$が与えられていると見なす．

なお，第3.3節で説明するように，直積法は述語分割$p(X_1, \ldots, X_m) \to p_1(X_{11}, \ldots, X_{1m_1}) \wedge \cdots \wedge p_h(X_{h1}, \ldots, X_{hm_h})$の述語に対し，$C_i, i = 1, \ldots, h$，を引数組$(X_{i1}, \ldots, X_{im_i})$のための定数組の集合として，$p[C_1 \times \cdots \times C_h]$の形をしたgaseを生成する．効率化のためには生成するgaseの総数を少なくする必要があるので，特に，多くの引数をもつ述語についてはすべて，$h \geq 2$である述語分割が与えられていることが望ましい．

本体にIDB述語を含まないルールを**初期化ルール**（initial rule），本体にIDB述語を含むルールを**主ルール**（main rule）と呼ぶ．主ルール

$$r: \quad s_0(\boldsymbol{X}'_0) \leftarrow A_1(\boldsymbol{Y}'_1), \ldots, A_l(\boldsymbol{Y}'_l), s_1(\boldsymbol{X}'_1), \ldots, s_n(\boldsymbol{X}'_n).$$

（ここで，$s_0, s_1, \ldots, s_n$はIDB述語，$A_1, \ldots, A_l$はEDB述語，$\boldsymbol{X}'_0, \boldsymbol{Y}'_1, \ldots, \boldsymbol{Y}'_l, \boldsymbol{X}'_1, \ldots, \boldsymbol{X}'_n$は引数ベクトル）に対し，上記の述語分割集合$PD$を使って，次のように無向グラフ$G_r$を作る．$G_r$の節点は，ルール$r$の各IDBアトム$s_0(\boldsymbol{X}'_0), s_1(\boldsymbol{X}'_1), \ldots, s_n(\boldsymbol{X}'_n)$を$PD$の中の述語分割にしたがって分割したときに得られるアトム，および，ルールrのEDBアトム$A_1(\boldsymbol{Y}'_1), \ldots, A_l(\boldsymbol{Y}'_l)$である．$G_r$の枝は，節点である二つのアトムが同じ変数を含むとき，それら二つの節点の間に枝を作る．この無向グラフG_rをルールrのための**引数依存グラフ**（argument dependency graph）と呼ぶ．

定義 3.1 主ルールの集合P_{main}と述語分割の集合PD（但し，上に述べたように，PDは実質的な述語分割を少なくとも一つ含むとする）が与えられ，P_{main}の各ルールrに対し引

数依存グラフG_rを作る．各引数依存グラフG_rが条件

1. $s_{0i}(\boldsymbol{X}'_{0i})$と$s_{0j}(\boldsymbol{X}'_{0j})$を頭部アトム$s_0(\boldsymbol{X}'_0)$の分割によりできた異なる節点とするとき，節点$s_{0i}(\boldsymbol{X}'_{0i})$と$s_{0j}(\boldsymbol{X}'_{0j})$の間に路が存在しない，

2. $s_{ki}(\boldsymbol{X}'_{ki})$と$s_{kj}(\boldsymbol{X}'_{kj})$を本体の同じIDBアトムの分割によりできた異なる節点とするとき，節点$s_{ki}(\boldsymbol{X}'_{ki})$と$s_{kj}(\boldsymbol{X}'_{kj})$の間に路が存在しない

を満たすとき，PDはP_{main}に対し**直積分割**（orthogonal decomposition）であると言う．datalog問題は，そのP_{main}に対して直積分割となるPDを少なくとも一つもつとき，**直積問題**（Cartesian product problem）であると言う．□

直積問題の条件の中に初期化ルールについての条件がないのは，初期化ルールより生成できる各gaを単純に一つのgaseとするからである．これらのgaseがgaseの初期値となる．例えば，初期化ルールよりga $p(c_1,c_2,c_3,c_4)$が生成でき，述語pの述語分割が$p(X_1,X_2,X_3,X_4)$ $\rightarrow p_1(X_1,X_2)\wedge p_2(X_3)\wedge p_3(X_4)$であるとすると，ga $p(c_1,c_2,c_3,c_4)$に対しgase $p[\{(c_1,c_2)\}\times\{c_3\}\times\{c_4\}]$を生成する．

主ルールrの各IDBアトム$s_i(\boldsymbol{X}'_i), i=0,1,\ldots,n,$を述語分割にしたがって分割したとき得られるアトムを$s_{i1}(\boldsymbol{X}'_{i1}),\ldots,s_{ih_i}(\boldsymbol{X}'_{ih_i})$と記す．直積問題の条件1は，ルール$r$から頭部アトム$s_0(\boldsymbol{X}'_0)$のためのgaseが生成できるための条件である．条件1が成立している場合，頭部アトム$s_0(\boldsymbol{X}'_0)$の引数のうち，引数$\boldsymbol{X}'_{0j}$部分の値は他の引数$\boldsymbol{X}'_{0k}, k\neq j,$部分の値に影響されない．故に，ルール$r$の本体に述語$s_1$のgaseと...述語$s_n$のgaseを代入して第3.1節の例で示したようなやり方で処理することにより（第3.3節で詳しく説明する），頭部アトムのためのgaseを生成することができる．

直積問題の条件2は，ルールrから頭部アトム$s_0(\boldsymbol{X}'_0)$のためのgaseを生成する際に，計算が効率的にできるための条件である．条件2が成立している場合，頭部アトム$s_0(\boldsymbol{X}'_0)$を分割したときに得られるいずれの引数部分$\boldsymbol{X}'_{0j}, j=1,\ldots,h_0,$も，本体の同じIDBアトム$s_i(\boldsymbol{X}'_i), i=1,\ldots,n,$を分割したときに得られる二つの以上の引数部分$\boldsymbol{X}'_{ij'},\ldots,\boldsymbol{X}'_{ik'}$に影響されることはない．故に，頭部アトムのためのgaseの計算において，本体に代入する$s_i, i=1,\ldots,n,$のgaseをそれが表す多数のgaに展開して代入する必要がない．この意味で，頭部アトムのためのgaseを効率的に計算することができる．

問題がdatalog問題ではない場合，意味IMPが無限集合になったり，IMPを定義すること自体が難しい（従って，解集合ANSを定義すること自体が難しい）場合がある（第

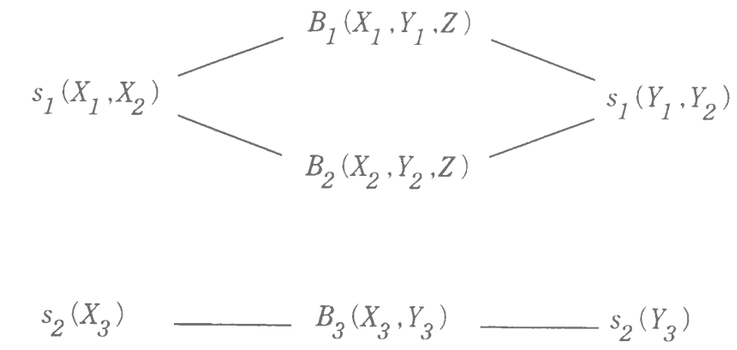


図3.1: 引数依存グラフG_1

1.4節を参照）．このような場合には，例え一つ一つのgaseを計算できても，すべてのgaseの計算を有限時間内に終えれなかったり，また，解集合を定義できないので，解集合を求めることができない．問題をdatalog問題に限定したのは，主にこれらの困難を避けるためである．問題がdatalog問題であり，かつ，上に述べた他の条件も満たすとき，すべてのgaseの計算を有限時間内に終え，かつ，それらのgaseから解集合を計算できることを保障することができる．

直積問題の例を示す．A,B_1,B_2,B_3をEDB述語，sをIDB述語として，ルール集合が

$$s(X_1,X_2,X_3)\leftarrow A(X_1,X_2,X_3). \tag{3.3}$$

$$s(X_1,X_2,X_3)\leftarrow B_1(X_1,Y_1,Z),B_2(X_2,Y_2,Z),B_3(X_3,Y_3),s(Y_1,Y_2,Y_3). \tag{3.4}$$

である問題（問い合わせアトムおよびファクト集合は省略．問題1と呼ぶ）を考える．ルール(3.3)は初期化ルール，ルール(3.4)は主ルールであり，主ルールはルール(3.4)だけである．述語分割集合$PD1$を$PD1=\{s(X_1,X_2,X_3)\rightarrow s_1(X_1,X_2)\wedge s_2(X_3)\}$として，主ルール(3.4)に対し引数依存グラフ$G_1$を作ると図3.1を得る．図3.1において，節点$s_1(X_1,X_2)$と節点$s_2(X_3)$は頭部アトム$s(X_1,X_2,X_3)$を分割してできた節点であり，節点$s_1(Y_1,Y_2)$と節点$s_2(Y_3)$は本体アトム$s(Y_1,Y_2,Y_3)$を分割してできた節点である．節点$s_1(X_1,X_2)$と節点$s_2(X_3)$の間に路がないので，グラフ$G_1$は条件1を満足する．また，節点$s_1(Y_1,Y_2)$と節点$s_2(Y_3)$の間に路がないので，$G_1$は条件2も満足する．故に，述語分割集合$PD1$は主ルール集合$\{(3.4)\}$に対し直積分割である．問題1は，datalog問題であり，かつ，直積分割$PD1$

をもつので，直積問題である．なお，$\{s(X_1,X_2,X_3) \rightarrow s_1(X_1) \wedge s_2(X_2) \wedge s_3(X_3)\}$ 等の $PD1$ 以外のいずれの述語分割集合も直積分割ではない．

直積問題でない例を示す．$A, B, C_1, C_2, C_3, C_4, D_1, D_2$ をEDB述語，p, q をIDB述語として，ルール集合が

$$p(X_1,X_2) \leftarrow A(X_1,X_2). \tag{3.5}$$

$$q(X_1,X_2) \leftarrow B(X_1,X_2). \tag{3.6}$$

$$p(X_1,X_5) \leftarrow C_1(X_1,X_2), C_2(X_1,X_3), C_3(X_2,X_4), C_4(X_3,X_4), p(X_2,X_3), q(X_3,X_5). \tag{3.7}$$

$$q(X_1,X_4) \leftarrow D_1(X_1,X_2), D_2(X_3,X_4), p(X_2,X_3). \tag{3.8}$$

である問題（問題2と呼ぶ）を考える．主ルールはルール(3.7)と(3.8)である．IDB述語 p, q の少なくとも一方を二つ以上に分割するような述語分割の集合には，$PD2 = \{p(X_1,X_2) \rightarrow p_1(X_1) \wedge p_2(X_2), q(X_1,X_2) \rightarrow q_1(X_1) \wedge q_2(X_2)\}$, $PD2' = \{p(X_1,X_2) \rightarrow p_1(X_1) \wedge p_2(X_2), q(X_1,X_2) \rightarrow q(X_1,X_2)\}$, $PD2'' = \{p(X_1,X_2) \rightarrow p(X_1,X_2), q(X_1,X_2) \rightarrow q_1(X_1) \wedge q_2(X_2)\}$ の三つがある．初めに $PD2$ について考える．$PD2$ を使って主ルール(3.7)に対し引数依存グラフ G_2 を作ると，本体アトム $p(X_2,X_3)$ を分割してできる二つの節点 $p_1(X_2)$ と $p_2(X_3)$ の間に路が存在するので，グラフ G_2 は条件2を満足しない．故に，$PD2$ は直積分割ではない．次に $PD2'$ について考える．$PD2'$ を使って主ルール(3.7)に対し引数依存グラフ G_3 を作ると，G_2 の場合と同様，本体アトム $p(X_2,X_3)$ を分割してできる二つの節点 $p_1(X_2)$ と $p_2(X_3)$ の間に路が存在するので，グラフ G_3 も条件2を満足しない．故に，$PD2'$ も直積分割ではない．最後に，$PD2''$ について考える．$PD2''$ を使って主ルール(3.8)に対し引数依存グラフ G_4 を作ると，頭部アトム $q(X_1,X_4)$ を分割してできる二つの節点 $q_1(X_1)$ と $q_2(X_4)$ の間に路が存在するので，グラフ G_4 は条件1を満足しない．故に，$PD2''$ は直積分割ではない．このように，いずれの分割集合 $PD2, PD2', PD2''$ も直積分割ではない．問題2は直積分割を持たないので，直積問題ではない．

直積問題クラスの広さについて，例を使って，説明する．ルール集合が

$$s(X,Y) \leftarrow A(X,Y). \tag{3.9}$$

$$s(X,Y) \leftarrow B(X,Z), s(Z,Y). \tag{3.10}$$

であり，問い合わせアトムが $s(a,Y)?$ である問題は典型的な右線形問題である [NRSU89b].

この問題は述語分割集合 $\{s(X,Y) \rightarrow s_1(X) \wedge s_2(Y)\}$ に関し直積問題になっている．この例のように，直積問題クラスは右線形問題クラスを包含している．ルール集合 $\{(3.20), (3.21)\}$ と問い合わせアトム $sg(e,Y)?$ をもつ第3.4節の問題は典型的な同世代問題である [BMSU86]. この問題は述語分割集合 $\{sg(X,Y) \rightarrow sg_1(X) \wedge sg_2(Y)\}$ に関し直積問題になっている．この例のように，直積問題クラスは同世代問題クラスを包含している．ルール集合 $P2$: $(3.38), (3.39),$ をもつ第3.5.1節の二つ目の問題は非線形問題である．この問題は述語分割集合 $\{s(X,Y) \rightarrow s_1(X) \wedge s_2(Y)\}$ に関し直積問題になっている．この例のように，直積問題クラスは一部の非線形問題を含んでいる．

3.3 直積法

直積法は，直積問題とその述語分割集合が与えられたとき，gaseを生成し，それらgaseの中から解集合を求める．述語 p のgaseにおける引数のグループ化の仕方は，述語 p の述語分割が示す引数のグループ化の仕方に等しい．すなわち，p の述語分割を

$$p(X_{11},\ldots,X_{1m_1},\ldots,X_{h1},\ldots,X_{hm_h}) \rightarrow p_1(X_{11},\ldots,X_{1m_1}) \wedge \cdots \wedge p_h(X_{h1},\ldots,X_{hm_h}) \tag{3.11}$$

とするとき，p のgaseは引数 $X_{11},\ldots X_{1m_1}$ のための定数組のある集合 $C_1 = \{(c_{11},\ldots c_{1m_1}), (c'_{11},\ldots c'_{1m_1}),\ldots\}$ と，…，引数 $X_{h1},\ldots X_{hm_h}$ のための定数組のある集合 $C_h = \{(c_{h1},\ldots c_{hm_h}), (c'_{h1},\ldots c'_{hm_h}),\ldots\}$ を使って，$p[C_1 \times \cdots \times C_h]$ と表される．以下，一般性を失うことなく，述語 p の述語分割では，式(3.11)のように，p の引数のある連続部分 $X_{i1},\ldots,X_{im_i}, i = 1,\ldots,h,$ が分割によりできる述語 p_i の引数になると仮定する．

直積法を図3.2に示す．以下，各stepごとに説明する．

(1) **step1**

初期化ルールより生成可能な異なる各ga $p(c_{11},\ldots,c_{1m_1},\ldots,c_{h1},\ldots,c_{hm_h})$ に対し，p の述語分割を式(3.11)として，gase $p[\{(c_{11},\ldots,c_{1m_1})\} \times \cdots \times \{(c_{h1},\ldots,c_{hm_h})\}]$ を生成する．step2で詳しく説明するように，直積法はgaseを二つのgase集合 NGS と OGS に分けて管理する．直積法は生成したgaseを，最初，NGS に保存する．

(2) **step2**

各主ルールの本体にstep1またはstep2で生成したgaseよりなるgase組を代入して，頭部アトムのためのgaseを生成していく．そして，この操作を，各主ルールの本体に，生成し

step1: /* gaseの初期値の生成 */
$OGS = \phi$;
$NGS = \bigcup \quad \{p[\{(c_{11}, \ldots, c_{1m_1})\} \times \cdots \times \{(c_{h1}, \ldots, c_{hm_h})\}]\}$;
初期化ルールより生成可能なga $p(c_{11}, \ldots, c_{1m_1}, \ldots, c_{h1}, \ldots, c_{hm_h})$
step2: /* gaseの生成 */
while($NGS \neq \phi$) {
step2.1: /* NGSの中のgaseを一つ選ぶ． */
$p[\boldsymbol{C}] = select\_gase(NGS)$;
step2.2: /* $p[\boldsymbol{C}]$を使ったgaseの生成 */
for (∀ルール$r \in P(p)$) {
/* $P(p)$は本体に述語pをもつルールの集合．ルールrの
頭部アトムを$p_0(\boldsymbol{X}_0)$，本体のIDBアトムを$p_1(\boldsymbol{X}_1), \ldots, p_n(\boldsymbol{X}_n)$と記す． */
while ($((p_1[\boldsymbol{C}_1], \ldots, p_n[\boldsymbol{C}_n]) = next\_gase\_tuple(r, p[\boldsymbol{C}], OGS)) \neq NULL$){
/* gase組$(p_1[\boldsymbol{C}_1], \ldots, p_n[\boldsymbol{C}_n])$の作成 */
if ($(p_0[C_{01} \times \cdots \times C_{0h}] = gene\_gase(r, (p_1[\boldsymbol{C}_1], \ldots, p_n[\boldsymbol{C}_n]))) \neq NULL$){
/* ルールrとgase組$(p_1[\boldsymbol{C}_1], \ldots, p_n[\boldsymbol{C}_n])$を使ったgaseの生成 */
if ($C_{01} \times \cdots \times C_{0h} \not\subseteq \bigcup_{p_0[C'_{01} \times \cdots \times C'_{0h}] \in NGS \cup OGS} C'_{01} \times \cdots \times C'_{0h}$) {
/* 被覆テスト */
step2.2.1: /* 不要なgaseの消去 */
$OGS = delete\_gase(p_0[C_{01} \times \cdots \times C_{0h}], OGS)$;
$NGS = delete\_gase(p_0[C_{01} \times \cdots \times C_{0h}], NGS)$;
step2.2.2:
$NGS = NGS \cup \{p_0[C_{01} \times \cdots \times C_{0h}]\}$;
} } } }
step2.3: /* $p[\boldsymbol{C}]$をNGSからOGSへ移す． */
if ($p[\boldsymbol{C}] \in NGS$) {
$NGS = NGS - \{p[\boldsymbol{C}]\}$;
$OGS = OGS \cup \{p[\boldsymbol{C}]\}$;
} }
step3: /* 解集合ANSの作成．問い合わせアトムを$q(\boldsymbol{X})?$と記す． */
$ANS = \phi$;
for (∀gase $q[\boldsymbol{C}^\dagger] \in OGS$) {
$GA = select\_ga(q(\boldsymbol{X})?, q[\boldsymbol{C}^\dagger])$;
$ANS = ANS \cup GA$;
}

図3.2: 直積法

たgaseよりなるすべてのgase組を代入し終わるまで繰り返す．

具体的には，次のように処理する．step1およびstep2を通して，生成したgaseの中で新しいgaを含むgase（すなわち，gase $p[\boldsymbol{C}]$が表すga集合がそれまでに生成したgaseが表すga集合の中にはない新しいgaを含むようなgase $p[\boldsymbol{C}]$）は保存し，そうでないgaseは不要であるので消去する．但し，step1で生成したgaseはすべて新しいgaを含んでいるので，(すでにstep1でNGSに保存したように）すべて保存する．保存するgaseは二つのgase集合NGSとOGSに分けて管理する．NGSは，そのgaseを含むいかなるgase組もまだルールの本体に代入したことがないようなgaseの集合であり，一方，OGSは，その中に含まれるgaseのみからなるいかなるgase組もすでに各ルールの本体に代入済みであるようなgaseの集合である．直積法は保存するgaseを，step1およびstep2を通して，最初，NGSの中に保存する．その後，step2で，NGSの中からgaseを一つ選び（$p[\boldsymbol{C}]$と記す），$p[\boldsymbol{C}]$とOGSの中の任意のgaseよりなるgase組で，少なくとも一カ所には$p[\boldsymbol{C}]$を含むようなすべてのgase組を作成して，それらgase組をルールの本体に代入してgaseを生成し，その後，$p[\boldsymbol{C}]$をNGSの中からOGSの中へ移す．step2において，このような処理を続けていくと，いずれは$NGS = \phi$となる．$NGS = \phi$はstep1またはstep2で生成したgaseよりなるすべてのgase組をすでにルールの本体に代入し終わったことを示すので，この時点で直積法はstep2を終了する．

step2の各stepを詳しく説明する．

(2.1) **step2.1**

サブルーチン$select_gase$はNGSの中のgaseを一つ選び，これを$p[\boldsymbol{C}]$として返す．具体的には，最も新しく生成されたgaseを選ぶ．

(2.2) **step2.2**

step2.2では，step2.1で選んだ$p[\boldsymbol{C}]$とOGSの中のgaseよりなるgase組を各主ルールの本体に代入してgaseを生成する．

step2.2の初めの部分について説明する．for(条件式){…}ブロックは，$p[\boldsymbol{C}]$の述語pを本体にもつ各主ルールr（頭部アトムを$p_0(\boldsymbol{X}_0)$と記し，アトム$p_1(\boldsymbol{X}_1), \ldots, p_n(\boldsymbol{X}_n)$の中の少なくとも一つは述語$p$のアトムとして，本体のIDBアトムを$p_1(\boldsymbol{X}_1), \ldots, p_n(\boldsymbol{X}_n)$と記す）に対し，while(条件式){…}ブロックを実行する．whileブロックの条件式が実行されると，そのたびに，サブルーチン$next_gase_tuple(r, p[\boldsymbol{C}], OGS)$は，ルール$r$の本体に代入するためのgase組として，gase $p[\boldsymbol{C}]$とOGSの中のgaseよりなるgase組で，少なく1箇所には$p[\boldsymbol{C}]$

を含むような新しいgase組を1組作ろうとする．$next_gase_tuple$は，そのようなgase組が作れる場合には，それを$(p_1[\boldsymbol{C}_1],\ldots,p_n[\boldsymbol{C}_n])$として返し，一方，作れない場合には$NULL$を返す（図3.2では，簡単のため，前者の場合を$(\ldots)\neq NULL$と記した）．前者の場合，続いて，1番目のif(条件式)$\{\cdots\}$ブロックを実行して，ルールrとgase組$(p_1[\boldsymbol{C}_1],\ldots,p_n[\boldsymbol{C}_n])$よりgaseを生成する．一方，後者の場合，このルールrのためのwhileブロックの実行を終了し，述語pを本体に含む次のルールのために同様の処理を行う．

step2.2の1番目のif(条件式)$\{\cdots\}$ブロックについて説明する．このifブロックの条件式が実行されると，サブルーチン$gene_gase(r,(p_1[\boldsymbol{C}_1],\ldots,p_n[\boldsymbol{C}_n]))$は，ルール$r$の本体にgase組$(p_1(\boldsymbol{C}_1),\ldots,p_n(\boldsymbol{C}_n))$を代入して，頭部アトムのためのgaseを生成しようとする．このgaseの生成の仕方を例を使って説明する．主ルールrを例えば

$$r:\quad p_0(X_1,X_2,X_3)\leftarrow A_1(X_1,X_2,Y_1,Z_2),A_2(X_3,Z_3),A_3(Y_2,Y_3,Z_1),\\ p_1(Y_1,Y_2,Y_3),p_2(Z_1,Z_2,Z_3).$$

とする．ここで，A_1,A_2,A_3はEDB述語，p_0,p_1,p_2はIDB述語であり，各IDB述語の述語分割は

$$p_0(X_1,X_2,X_3)\rightarrow p_{01}(X_1,X_2)\wedge p_{02}(X_3),$$
$$p_1(Y_1,Y_2,Y_3)\rightarrow p_{11}(Y_1)\wedge p_{12}(Y_2,Y_3),$$
$$p_2(Z_1,Z_2,Z_3)\rightarrow p_{21}(Z_1)\wedge p_{22}(Z_2)\wedge p_{23}(Z_3)$$

とする．また，ルールrの本体に代入するgase組を，$C_{11},C_{21},C_{22},C_{23}$を定数の集合，$C_{12}$を二つの定数よりなる定数組の集合として，$(p_1[C_{11}\times C_{12}],p_2[C_{21}\times C_{22}\times C_{23}])$と記す．ルール$r$より引数依存グラフを作り，路で接続された部分を取り出すと，次の三つの部分が得られる．

$$p_{01}(X_1,X_2)\leftarrow A_1(X_1,X_2,Y_1,Z_2),p_{11}(Y_1),p_{22}(Z_2).\tag{3.12}$$
$$p_{02}(X_3)\leftarrow A_2(X_3,Z_3),p_{23}(Z_3).\tag{3.13}$$
$$A_3(Y_2,Y_3,Z_1)\wedge p_{12}(Y_2,Y_3)\wedge p_{21}(Z_1).\tag{3.14}$$

ここで，ルールrの頭部アトム$p_0(X_1,X_2,X_3)$より作られるアトム$p_{01}(X_1,X_2)$と$p_{02}(X_3)$は$\leftarrow$の左側に記した．この三つの式と集合$C_{11},C_{12},C_{21},C_{22},C_{23}$を使って，ルール$r$の頭部アトム$p_0(X_1,X_2,X_3)$のためのgaseを次のように生成する．式(3.12)を恰も，EDB述語A_1と，

ファクト集合が$\{p_{11}(c)\mid c\in C_{11}\}$であるEDB述語$p_{11}$と，ファクト集合が$\{p_{22}(c)\mid c\in C_{22}\}$であるEDB述語$p_{22}$を本体にもつルールと見なし，これらのルール(3.12)とA_1ファクト集合とp_{11}ファクト集合とp_{22}ファクト集合より，生成可能なすべてのp_{01} gaを生成する（処理1と呼ぶ）．それらのp_{01} gaの集合を$\{p_{01}(c_1,c_2)\mid(c_1,c_2)\in C_{01}\}$と記す．式(3.13)についても同様に処理して生成可能なすべてのp_{02} gaを生成し（処理2と呼ぶ），それらのp_{02} gaの集合を$\{p_{02}(c_3)\mid c_3\in C_{02}\}$と記す．また，式(3.14)については，$A_3$ファクトと，ga集合$\{p_{12}(c_2,c_3)\mid(c_2,c_3)\in C_{12}\}$の中の$p_{11}$ gaと，ga集合$\{p_{21}(c_1)\mid c_1\in C_{21}\}$の中の$p_{21}$ gaよりなるga組で式(3.14)を真とするようなga組$(A_1(c_2,c_3,c_1),p_{12}(c_2,c_3),p_{21}(c_1))$が少なくとも一つ存在するか否かを調べる（処理3と呼ぶ）．これらの処理の結果，$\{p_{01}(c_1,c_2)\mid(c_1,c_2)\in C_{01}\}\neq\phi$，かつ，$\{p_{02}(c_3)\mid c_3\in C_{02}\}\neq\phi$，かつ，式(3.14)を真とするようなga組$(A_1(c_2,c_3,c_1),p_{12}(c_2,c_3),p_{21}(c_1))$が少なくとも一つ存在する場合，ルール$r$の頭部アトム$p_0(X_1,X_2,X_3)$のためのgaseとして$p_0[C_{01}\times C_{02}]$を生成する．一方，そうでない場合，ルール$r$の頭部アトム$p_0(X_1,X_2,X_3)$のためのgaseの生成は失敗する．

なお，処理1および処理2は本体がEDB述語のみからなるルールよりgaを生成する処理であり，このような処理を効率的に行う方法が関係データベースの分野において数多く提案されている[Ull88, Ull89]．また，処理3は論理式を真とするga組の存在を判定する処理であるが，このような処理も，関係データベースの分野において提案された同じ方法を少し変更することにより，効率的に行うことができる．故に，サブルーチン$gene_gase$は，関係データベースの分野において提案された方法を利用して，処理1,2,3を効率的に行う．

式(3.12)，(3.13)，(3.14)のように，もとのルールrの分割によりできる式を**部分ルール**(partial rule）と呼び，特に，式(3.12)，(3.13)のように，gaを生成するためのものを**生成型部分ルール**（partial rule for generation），式(3.14)のように，ga組の存在を判定するためのものを**判定型部分ルール**（partial rule for decision）と呼ぶ．

サブルーチン$gene_gase(r,(p_1[\boldsymbol{C}_1],\ldots,p_n[\boldsymbol{C}_n]))$は，頭部アトムのためのgaseの生成に成功した場合は，それを$p_0[C_{01}\times\cdots\times C_{0h}]$として返し，一方，失敗した場合は$NULL$を返す(図3.2では，簡単のため，前者の場合を$(\ldots)\neq NULL$と記した）．前者の場合，続いて，2番目のif(条件式)$\{\cdots\}$ブロックを実行し，一方，後者の場合，このgase組$(p_1[\boldsymbol{C}_1],\ldots,p_n[\boldsymbol{C}_n])$のための1番目のifブロックの実行を終了し，whileブロックの条件式，すなわち，サブルーチン$next_gase_tuple$の実行によりルールrのための次のgase組を生成し，そのgase組を使って同様の処理を行う．

step2.2の2番目のif(条件式){…}ブロックについて説明する．このifブロックの条件式では

$$C_{01} \times \cdots \times C_{0h} \not\subseteq \bigcup_{p_0[C'_{01} \times \cdots \times C'_{0h}] \in NGS \cup OGS} C'_{01} \times \cdots \times C'_{0h} \tag{3.15}$$

が成立するか否かを判定する．すなわち，gase $p_0[C_{01} \times \cdots \times C_{0h}]$ が表すga集合の中に，step1またはstep2でこれまでに生成されたgaseが表すga集合の中にはない新しいgaが含まれているか否かを判定する．この判定検査を**被覆テスト**と呼ぶ．

被覆テストは効率を考慮して次のように行う．NGS または OGS に含まれる述語 p_0 のgaseを $p_0[C^1_{01} \times \cdots \times C^1_{0h}], p_0[C^2_{01} \times \cdots \times C^2_{0h}], \ldots, p_0[C^f_{01} \times \cdots \times C^f_{0h}]$ と記す．式(3.15)の判定は，式

$$C_{01} \times \cdots \times C_{0h} - C^1_{01} \times \cdots \times C^1_{0h} - C^2_{01} \times \cdots \times C^2_{0h} - \cdots - C^f_{01} \times \cdots \times C^f_{0h} \tag{3.16}$$

の値が空集合でないか空集合であるかを判定することにより行う．h 個の集合よりなる直積 $C_1 \times C_2 \times C_3 \times \cdots \times C_h$ と $C'_1 \times C'_2 \times C'_3 \times \cdots \times C'_h$ の差 $C_1 \times \cdots \times C_h - C'_1 \times \cdots \times C'_h$ は，二つの直積が共通部分をもつとき，式

$$\begin{aligned} &C_1 \times C_2 \times C_3 \times \cdots \times C_h - C'_1 \times C'_2 \times C'_3 \times \cdots \times C'_h = \\ &\quad (C_1 - C'_1) \times C_2 \times C_3 \times \cdots \times C_h \cup (C_1 \cap C'_1) \times (C_2 - C'_2) \times C_3 \times \cdots \times C_h \\ &\quad \cup \cdots \cup (C_1 \cap C'_1) \times (C_2 \cap C'_2) \times \cdots \times (C_{h-1} \cap C'_{h-1}) \times (C_h - C'_h) \end{aligned} \tag{3.17}$$

によって，高々h個の直積の和として表すことができる．ここで，$C_j \subseteq C'_j, j = 1, \ldots, h,$ の場合，そのような集合の差 $C_j - C'_j$ を含む直積 $(C_1 \cap C'_1) \times \cdots \times (C_j - C'_j) \times \cdots \times C_h$ は存在しない．直積法は，直積のこの性質を利用して，式(3.16)の値を次のように計算する．$C^1_{01} \times \cdots \times C^1_{0h}, \ldots, C^f_{01} \times \cdots \times C^f_{0h}$ の中から $C_{01} \times \cdots \times C_{0h}$ と共通部分をもつ最初の直積（すなわち，共通部分をもつ直積で右上の番号が最小の直積）を探し（それをi番目の $C^i_{01} \times \cdots \times C^i_{0h}$ とする），式(3.17)にしたがって，$C_{01} \times \cdots \times C_{0h} - C^i_{01} \times \cdots \times C^i_{0h}$ を計算して直積 $(C_{01} - C^i_{01}) \times \cdots \times C_{0h}, \ldots, (C_{01} \cap C^i_{01}) \times \cdots \times (C_{0h} - C^i_{0h})$ を作成する $(C_{01} \times \cdots \times C_{0h} - C^i_{01} \times \cdots \times C^i_{0h} = (C_{01} - C^i_{01}) \times \cdots \times C_{0h} \cup \cdots \cup (C_{01} \cap C^i_{01}) \times \cdots \times (C_{0h} - C^i_{0h}))$．作成したこれらの直積を**作業用直積**と呼ぶ．次に，作成した作業用直積 $(C_{01} - C^i_{01}) \times \cdots \times C_{0h}, \ldots, (C_{01} \cap C^i_{01}) \times \cdots \times (C_{0h} - C^i_{0h})$ の各々に対し，$C^{i+1}_{01} \times \cdots \times C^{i+1}_{0h}, \ldots, C^f_{01} \times \cdots \times C^f_{0h}$ の中から共通部分をもつ最初の直積を探し，上と同様に，それを引いて作業用直積を作成

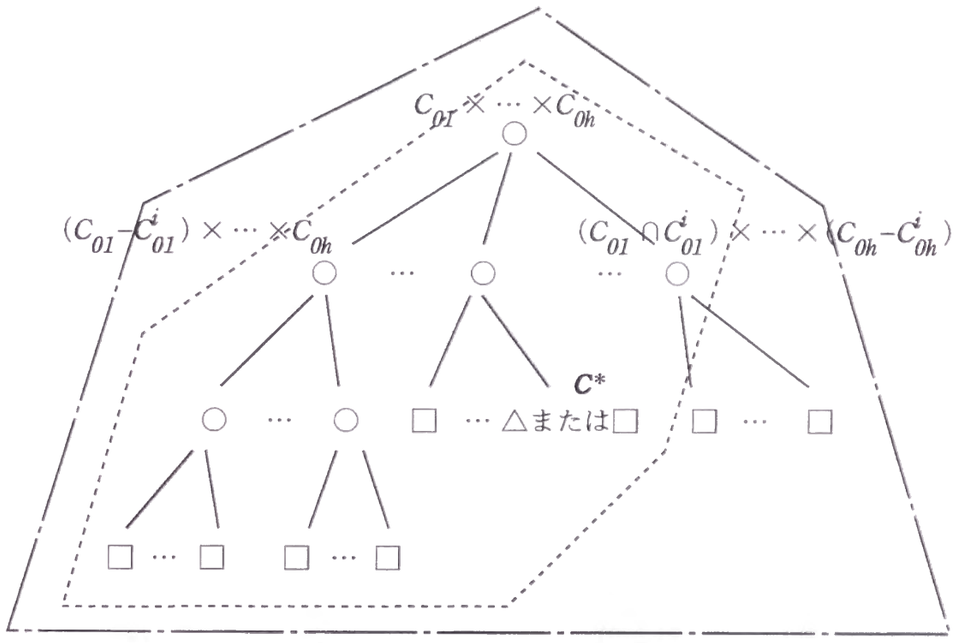


図3.3: 被覆テストにおける計算を表す図

する．このような処理を繰り返すと，図3.3に示すように，作業用直積が次々と生成される．図3.3において，根は直積$C_{01}\times\cdots\times C_{0h}$であり，他の節点は計算の途中で生成された作業用直積である．□の葉節点$\boldsymbol{C}$はある直積$\boldsymbol{C}''$（$\in\{C_{01}^1\times\cdots\times C_{0h}^1,\ldots,C_{01}^f\times\cdots\times C_{0h}^f\}$）の減算$\boldsymbol{C}-\boldsymbol{C}''$の結果，$\boldsymbol{C}\subseteq\boldsymbol{C}''$であるために，完全に消去されてしまった作業用直積であり，一方，△の葉節点$\boldsymbol{C}'$は，$\boldsymbol{C}'$と共通部分をもつ直積$\boldsymbol{C}''$（$\in\{C_{01}^1\times\cdots\times C_{0h}^1,\ldots,C_{01}^f\times\cdots\times C_{0h}^f\}$）が存在しないために減算を行うことができず，消去されずに残ってしまった作業用直積である．節点$\boldsymbol{C}^*$が△の場合のように，△の葉節点が一つでも見つかった時点で，直積法は以後の計算は打ち切り（すなわち，図の点線内の木だけが生成し），式(3.16)の値は空集合でないと判定する．一方，節点$\boldsymbol{C}^*$が□の場合のように，△の葉節点が最後まで見つからない場合は，直積法は葉節点がすべて□である一点鎖線内の木を生成した後に，式(3.16)の値は空集合であると判定する．

なお，直積$C_{01}\times\cdots\times C_{0h}$や作業用直積と共通部分をもつ直積の探索は，単純には，直積$C_{01}^1\times\cdots\times C_{0h}^1,\ldots,C_{01}^f\times\cdots\times C_{0h}^f$を順番に一つ一つ調べていくことにより行える．しかし，このやり方は非効率であるため，直積法は次のように探索を行う．直積法は，step1またはstep2でgaseを生成するときに，gaseに対し**識別番号**を昇順に付けていき，そして，この識別番号によってNGSまたはOGSの中のgaseに直接アクセスできるようにしている．説明の簡単のため，gase（$p_0[C_1\times\cdots\times C_h]$と記す）のいずれの集合$C_j,j=1,\ldots,h,$も定数の集合である（$C_j=\{c_1,c_2,\ldots\}$）と仮定する．述語$p_0$，整数$j$，定数$c$に対し，$NGS$または$OGS$の中の述語$p_0$のgaseで，その$j$番目の集合$C_j$が定数$c$を含む（$c\in C_j$）ようなgase $p_0[C_1\times\cdots\times C_j\times\cdots\times C_h]$を考える．直積法は，各組$(p_0,j,c)$に対し，このようなgase $p_0[C_1\times\cdots\times C_j\times\cdots\times C_h]$の**識別番号の集合**（$ID(p_0,j,c)$と記す）を管理しており，そして，各集合$ID(p_0,j,c)$を，識別番号を降順に並べたリストとして記憶している．直積$C_{01}\times\cdots\times C_{0h},C_{01}^1\times\cdots\times C_{0h}^1,\ldots,C_{01}^f\times\cdots\times C_{0h}^f$を上の段落の場合と同様に定義する．但し，直積$C_{01}^1\times\cdots\times C_{0h}^1,\ldots,C_{01}^f\times\cdots\times C_{0h}^f$は，対応するgaseの識別番号の降順に並べておく．初めに，$C_{01}^1\times\cdots\times C_{0h}^1,\ldots,C_{01}^f\times\cdots\times C_{0h}^f$の中から，$C_{01}\times\cdots\times C_{0h}$と共通部分をもつ最初の直積を探す場合を考える．このような直積に対応するgaseの識別番号は，集合$ID(p_0,j,c)$の定義より，集合$\cap_{j=1,\ldots,h}(\cup_{c\in C_{0j}}ID(p_0,j,c))$の中の最大要素（$id_0$と記す）である．集合$ID(p_0,j,c)$は降順に並んだ識別番号のリストであるので，直積法は，リスト$ID(p_0,j,c),j=1,\ldots,h,c\in C_{0j},$の要素を先頭から順に調べていくことによって，$id_0$を効率的に探す（$ID(p_0,j,c)$の中の$k_1$未満$k_2$以上の要素の数を$cnt(p_0,j,c,k_1,k_2)$と記すと，$O(\sum_{j=1,\ldots,h,c\in C_{0j}}cnt(p_0,j,c,\infty,id_0))$の時間量で$id_0$を見つけることができる）．次に，上で見つかった識別番号id_0のgaseの直積がi番目の$C_{01}^i\times\cdots\times C_{0h}^i$であるとし，更に，$C_{01}\times\cdots\times C_{0h}-C_{01}^i\times\cdots\times C_{0h}^i$の計算により作成される作業用直積の一つを$C_1'\times\cdots\times C_h'$と記して，残りの直積$C_{01}^{i+1}\times\cdots\times C_{0h}^{i+1},\ldots,C_{01}^f\times\cdots\times C_{0h}^f$の中から，$C_1'\times\cdots\times C_h'$と共通部分をもつ最初の直積を探す場合を考える．集合$ID(p_0,j,c)$の中のid_0未満の要素よりできる集合を$ID_<(p_0,j,c,id_0)$と記す．このとき，求める直積に対応するgaseの識別番号は，集合$\cap_{j=1,\ldots,h}(\cup_{c\in C_j'}ID_<(p_0,j,c,id_0))$の中の最大要素（$id_1$と記す）である．従って，直積法は，$id_0$を求めたときの識別番号集合の一部$ID(p_0,j,c),j=1,\ldots,h,c\in C_j'$（ここで，$C_j'\subseteq C_{0j}$），について，$ID(p_0,j,c)$の要素を先ほど調べ終えた位置から後方に向かって引き続き調べていくことによって，id_1を効率的に探す（$O(\sum_{j=1,\ldots,h,c\in C_j'}cnt(p_0,j,c,id_0,id_1))$の時間量で$id_1$を見つけることができる）．直積法は集合$ID(p_0,j,c)$に対してこのような処理を繰り返すことにより，直積$C_{01}\times\cdots\times C_{0h}$や作業用直積と共通部分をもつ直積を効率的に探索していく．

識別番号集合$ID(p_0,j,c)$に関連して説明を補足する．$ID(p_0,j,c)$は，gase集合$NGS\cup OGS$にgaseが追加されたり削除されたりする度に更新する必要がある．gaseを削除する際の更新を効率的に行うために，直積法は，$NGS\cup OGS$の中の各gaseの識別番号idに対し，集合$ID(p_0,j,c)$を表すリストの中の要素idに直接アクセスできるように，その要素への**ポインタの集合**$PTR(id)=\{ptr_1,ptr_2,\ldots\}$を管理している．

以上のようにstep2.2の被覆テストを実行した結果，式(3.15)が成立する場合，すなわち，gase $p_0[C_{01}\times\cdots\times C_{0h}]$が新しいgaを含んでいる場合，step2.2.1およびstep2.2.2を実行する．一方，式(3.15)が成立しない場合，すなわち，gase $p_0[C_{01}\times\cdots\times C_{0h}]$が新しいgaを含んでいない場合，$p_0[C_{01}\times\cdots\times C_{0h}]$を捨てて，2番目のifブロックおよび1番目のifブロックの実行を終了し，再び，step2.2のwhileブロックの条件式を実行して新しいgase組を生成し，そのgase組を使ってこれまでと同様の処理を行う．

step2.2.1およびstep2.2.2について説明する．step2.2.2ではgase $p_0[C_{01}\times\cdots\times C_{0h}]$をgase集合$NGS$に保存する．また，それに先だって，step2.2.1では，OGSおよびNGSの中のgaseを調べ，$p_0[C_{01}\times\cdots\times C_{0h}]$に包含されるgaseがあれば，それを消去する（サブルーチン*delete_gase*はこれらの処理を表す）．

$p_0[C_{01}\times\cdots\times C_{0h}]$に包含されるgase（$p_0[C_1\times\cdots\times C_h]$と記す）を消去するのは次の理由による．(i) $p_0[C_1\times\cdots\times C_h]$を含むgase組をルールの本体に代入して生成されるgase

$p[C'_1 \times \cdots \times C'_l]$ は，その gase 組の gase $p_0[C_1 \times \cdots \times C_h]$ を gase $p_0[C_{01} \times \cdots \times C_{0h}]$ に置き換えて得られる gase 組を同じルールの本体に代入して生成される gase $p[C''_1 \times \cdots \times C''_l]$ に包含される．それ故，正しい解集合を得るためには，gase $p_0[C_{01} \times \cdots \times C_{0h}]$ を保存すれば，gase $p_0[C_1 \times \cdots \times C_h]$ は不要である．また，(ii) 効率化のためには，通常，不要な gase 組を作らない方がよく，そのためには，保存しておく gase の数を減らす方がよい．以上の二つが理由である．

なお，OGS（NGS）の中から $p_0[C_{01} \times \cdots \times C_{0h}]$ に包含される gase を探す処理は，OGS（NGS）の中の gase（$p_0[C'_1 \times \cdots \times C'_h]$ と記す）を一つ一つ調べて，$C'_1 \subseteq C_{01} \wedge \cdots \wedge C'_h \subseteq C_{0h}$ が成立するか否かを判定することにより行う．ここで，集合 $C_{0i}, i = 1, \ldots, h,$ の要素は予め整列しておき，各集合の包含関係 $C'_i \subseteq C_{0i}, i = 1, \ldots, h,$ の判定は，C'_i の中の各要素 c について，c が C_{0i} に含まれる否かを判定していくことにより行う（c が C_{0i} に含まれる否かの判定は，各 c に対し，$O(1)$ 程度の時間量で行える）．

(2.3) **step2.3**

step2.1 で選択された gase $p[\boldsymbol{C}]$ を NGS の中から OGS の中に移す．

(3) **step3**

step3 では，OGS の中の gase より解集合 ANS を求める．

問い合わせアトムを $q(\boldsymbol{X})?$ と記す．ここで，$\boldsymbol{X}$ は定数および/または変数よりなる任意の引数ベクトルである．step3 の開始時点では，OGS の中に保存されている gase が表す ga 集合の和は，問題から生成可能なすべての ga の集合（すなわち，問題の ddb の意味）に等しくなっている．故に，直積法は OGS の中の述語 q の各 gase（$q[\boldsymbol{C}^\dagger]$ と記す）に対し，サブルーチン $select\_ga(q(\boldsymbol{X})?, q[\boldsymbol{C}^\dagger])$ を使って，$q[\boldsymbol{C}^\dagger]$ が表す ga 集合の中の ga で，問い合わせアトム $q(\boldsymbol{X})?$ に適合するような ga の集合 $\{q(\boldsymbol{c}) \mid \boldsymbol{c} \in \boldsymbol{C}^\dagger,$ かつ，$q(\boldsymbol{c})$ は $q(\boldsymbol{X})?$ に適合する $\}$（$\{q(\boldsymbol{c}) \mid \cdots\}$ と略記する）を計算し，そして，それら ga 集合の和として解集合 ANS を求める．

サブルーチン $select\_ga$ について詳しく説明する．ga 集合 $\{q(\boldsymbol{c}) \mid \cdots\}$ の計算は，単純には，gase $q[\boldsymbol{C}^\dagger]$ が表す ga をすべて生成し，それらの ga の中から問い合わせアトム $q(\boldsymbol{X})?$ に適合する ga を選択することにより，行うことができる．しかし，このやり方は，解ではない多数の ga を生成するため，非効率である．そのため，サブルーチン $select\_ga(q(\boldsymbol{X})?, q[\boldsymbol{C}^\dagger])$ は，ルールから ga を生成する方法を使って ga 集合 $\{q(\boldsymbol{c}) \mid \cdots\}$ を効率的に計算する．

例を使って説明する．問い合わせアトム $q(\boldsymbol{X})?$ を $q(a, X, X, Y)?$，述語 q の述語分割を

$$q(X_1, X_2, X_3, X_4) \rightarrow q_1(X_1, X_2) \wedge q_2(X_3, X_4). \tag{3.18}$$

C_1 と C_2 を二つの定数よりなる定数組の集合として，gase $q[\boldsymbol{C}^\dagger]$ を $q[C_1 \times C_2]$ と記す．問い合わせアトム $q(a, X, X, Y)?$ を式 (3.18) のアトム $q(X_1, X_2, X_3, X_4)$ に代入し，できた式の $\rightarrow$ を $\leftarrow$ に変更すると，式

$$q(a, X, X, Y) \leftarrow q_1(a, X) \wedge q_2(X, Y). \tag{3.19}$$

を得る．式 (3.19) を，恰も，ファクト集合が $\{q_1(c_1, c_2) \mid (c_1, c_2) \in C_1\}$ である EDB 述語 q_1 と，ファクト集合が $\{q_2(c_1, c_2) \mid (c_1, c_2) \in C_2\}$ である EDB 述語 q_2 を本体にもつルールと見なし，これらのルールとファクト集合より生成可能なすべての q ga を生成すると，これらの q ga の集合は求めるべき ga 集合 $\{q(\boldsymbol{c}) \mid \cdots\} = \{q(c'_1, c'_2, c'_3, c'_4) \mid (c'_1, c'_2, c'_3, c'_4) \in C_1 \times C_2,$ かつ，$q(c'_1, c'_2, c'_3, c'_4)$ は $q(a, X, X, Y)?$ に適合する $\}$ に一致する．

本体が EDB 述語のみからなるルールより ga を効率的に生成する方法は，先に述べたように，関係データベースの分野において数多く提案されている．従って，サブルーチン $select\_ga$ は，これらの方法を使って，求めるべき ga 集合 $\{q(c'_1, c'_2, c'_3, c'_4) \mid (c'_1, c'_2, c'_3, c'_4) \in C_1 \times C_2,$ かつ，$q(c'_1, c'_2, c'_3, c'_4)$ は $q(a, X, X, Y)?$ に適合する $\}$ を効率的に計算する．

式 (3.19) のように，問い合わせアトムの分割によりできる式を**解ルール**（answer rule）と呼ぶ．

3.4 適用例

直積法の適用例を示す．ルール集合

$$P: \quad sg(X, X) \leftarrow A(X). \tag{3.20}$$

$$sg(X, Y) \leftarrow B(X', X), B(Y', Y), sg(X', Y'). \tag{3.21}$$

とファクト集合

$$\begin{aligned} D \;=\; & \{A(a), A(b), A(c), A(d), A(e), A(f), A(g)\} \\ & \cup \{B(a, c), B(b, c), B(a, d), B(b, d), B(c, e), B(c, f), B(d, f), B(d, g)\} \end{aligned} \tag{3.22}$$

と問い合わせアトム $sg(e,Y)?$ よりなる問題 $(sg(e,Y)?,(P,D))$ を考える．この問題は，第3.2節で述べたように，述語分割集合 $PD=\{sg(X,Y)\rightarrow sg_1(X)\wedge sg_2(Y)\}$ に対し直積問題である．この問題に直積法を適用する．

step1で，初期化ルール(3.20)よりgaseを生成すると，

$$\begin{aligned} NGS &= \{sg[\{a\}\times\{a\}], sg[\{b\}\times\{b\}], sg[\{c\}\times\{c\}], sg[\{d\}\times\{d\}], sg[\{e\}\times\{e\}], \\ &\quad sg[\{f\}\times\{f\}], sg[\{g\}\times\{g\}]\} \end{aligned} \tag{3.23}$$

を得る．

step2.1で，$p[\boldsymbol{C}]$ として，NGS の中のgase $sg[\{a\}\times\{a\}]$ を選ぶ．

このgase $sg[\{a\}\times\{a\}]$ に対するstep2.2およびstep2.3の処理を説明する．この問題では，主ルールはルール(3.21)一つだけであるので，step2.2のforブロックの条件式で選ばれる主ルールはこのルール一つだけである．また，ルール(3.21)は線形であるので，サブルーチン *next_gase_tuple* が各gase $p[\boldsymbol{C}]$ に対して作るgase組は $(p[\boldsymbol{C}])$ 一つだけである．今，$p[\boldsymbol{C}]=sg[\{a\}\times\{a\}]$ であるので，サブルーチン *next_gase_tuple* はgase組 $(sg[\{a\}\times\{a\}])$ を作成する．サブルーチン *gene_gase* はこのgase組をルール(3.21)の本体に代入してgaseを生成しようとする．ルール(3.21)を sg の述語分割にしたがって分割すると，

$$sg_1(X)\leftarrow B(X',X), sg_1(X'). \tag{3.24}$$

$$sg_2(Y)\leftarrow B(Y',Y), sg_2(Y'). \tag{3.25}$$

の二つの部分ルールが得られる．部分ルール(3.24)と，gase $sg[\{a\}\times\{a\}]$ の1番目の定数集合 $\{a\}$ から得られるga集合 $\{sg_1(a)\}$ と，式(3.22)の B ファクト集合からgaを生成すると，ga集合 $\{sg_1(c), sg_1(d)\}$ を得る．同様に，部分ルール(3.25)と，gase $sg[\{a\}\times\{a\}]$ の2番目の定数集合 $\{a\}$ から得られるga集合 $\{sg_2(a)\}$ と，B ファクト集合からgaを生成すると，ga集合 $\{sg_2(c), sg_2(d)\}$ を得る．よって，*gene_gase* はgase $sg[\{c,d\}\times\{c,d\}]$ を生成する．被覆テストにおいて，このgaseの直積 $\{c,d\}\times\{c,d\}$ からgase集合 NGS（式(3.23)）および OGS（但し，空）の中のgaseの直積を引くと，

$$\begin{aligned} &\{c,d\}\times\{c,d\}-\{a\}\times\{a\}-\cdots-\{g\}\times\{g\} \\ &= (\{d\}\times\{c,d\}\cup\{c\}\times\{d\})-\{d\}\times\{d\}-\cdots-\{g\}\times\{g\} \\ &= (\{d\}\times\{c\}-\{e\}\times\{e\}-\cdots-\{g\}\times\{g\}) \end{aligned} \tag{3.26}$$

$$\cup(\{c\}\times\{d\}-\{d\}\times\{d\}-\cdots-\{g\}\times\{g\})$$
$$= \{d\}\times\{c\}\cup(\{c\}\times\{d\}-\{d\}\times\{d\}-\cdots-\{g\}\times\{g\})$$

のように計算が進み，少なくとも $\{d\}\times\{c\}$ が残り，式(3.26)が空集合ではないことが分かる．すなわち，gase $sg[\{c,d\}\times\{c,d\}]$ が新しいgaを含んでいることが分かる．従って，step2.2.1で，このgase $sg[\{c,d\}\times\{c,d\}]$ に包含される，NGS および OGS の中のgase（すなわち，gase $sg[\{c\}\times\{c\}]$ と $sg[\{d\}\times\{d\}]$）を消去し，その後，step2.2.2でこのgaseを NGS に加える．step2.2を終了し，step2.3でgase $sg[\{a\}\times\{a\}]$ を NGS から OGS へ移す．この結果，

$$\begin{aligned} NGS &= \{sg[\{c,d\}\times\{c,d\}], sg[\{b\}\times\{b\}], sg[\{e\}\times\{e\}], sg[\{f\}\times\{f\}], \\ &\quad sg[\{g\}\times\{g\}]\}, \end{aligned} \tag{3.27}$$

$$OGS = \{sg[\{a\}\times\{a\}]\} \tag{3.28}$$

となる．

step2の初めに戻り，step2.1で NGS（式(3.27)）の中から次の $p[\boldsymbol{C}]$ としてgase $sg[\{c,d\}\times\{c,d\}]$ を選び，このgaseのために，再びstep2.2およびstep2.3を実行する．step2.2で，*next_gase_tuple* はgase組 $(sg[\{c,d\}\times\{c,d\}])$ を作成し，*gene_gase* はこのgase組のためにgase $sg[\{e,f,g\}\times\{e,f,g\}]$ を生成する．被覆テストで，このgase $sg[\{e,f,g\}\times\{e,f,g\}]$ の直積 $\{e,f,g\}\times\{e,f,g\}$ から，NGS（式(3.27)）および OGS（式(3.28)）の中のgaseの直積を引くと

$$\{e,f,g\}\times\{e,f,g\}-\{c,d\}\times\{c,d\}-\{a\}\times\{a\}-\cdots-\{g\}\times\{g\}$$

が空集合ではないことが分かる．従って，step2.2.1で，このgase $sg[\{e,f,g\}\times\{e,f,g\}]$ に包含される，NGS および OGS の中のgase（すなわち，gase $sg[\{e\}\times\{e\}]$ と $sg[\{f\}\times\{f\}]$ と $sg[\{g\}\times\{g\}]$）を消去し，その後，step2.2.2でこのgaseを NGS に加える．step2.3でgase $sg[\{c,d\}\times\{c,d\}]$ を NGS から OGS へ移した後，

$$NGS = \{sg[\{e,f,g\}\times\{e,f,g\}], sg[\{b\}\times\{b\}]\}, \tag{3.29}$$

$$OGS = \{sg[\{c,d\}\times\{c,d\}], sg[\{a\}\times\{a\}]\} \tag{3.30}$$

となる．

再び，step2の初めに戻り，step2.1で$p[\boldsymbol{C}]$として，gase $sg[\{e,f,g\}\times\{e,f,g\}]$を選び，step2.2およびstep2.3を実行する．step2.2において，$gene_gase$はgase組$(sg[\{e,f,g\}\times\{e,f,g\}])$のためのgaseを生成しようとするが，適切な$B$ファクトがないためにgaseの生成に失敗する．従って，step2.3でgase $sg[\{e,f,g\}\times\{e,f,g\}]$を$NGS$から$OGS$へ移した後，

$$NGS = \{sg[\{b\}\times\{b\}]\}, \tag{3.31}$$

$$OGS = \{sg[\{e,f,g\}\times\{e,f,g\}], sg[\{c,d\}\times\{c,d\}], sg[\{a\}\times\{a\}]\} \tag{3.32}$$

となる．

再度，step2の初めに戻り，step2.1で$p[\boldsymbol{C}]$としてgase $sg[\{b\}\times\{b\}]$を選び，step2.2およびstep2.3を実行する．step2.2で$gene_gase$はgase組$(sg[\{b\}\times\{b\}])$のためにgase $sg[\{c,d\}\times\{c,d\}]$を生成する．しかし，被覆テストにおいて

$$\{c,d\}\times\{c,d\}-\{e,f,g\}\times\{e,f,g\}-\{c,d\}\times\{c,d\}-\{a\}\times\{a\}-\{b\}\times\{b\}$$

が空集合であることが分かるので，gase $sg[\{c,d\}\times\{c,d\}]$を保存することなく捨てる．step2.3でgase $sg[\{b\}\times\{b\}]$をNGSからOGSへ移した後，

$$NGS = \phi, \tag{3.33}$$

$$OGS = \{sg[\{b\}\times\{b\}], sg[\{e,f,g\}\times\{e,f,g\}], sg[\{c,d\}\times\{c,d\}], sg[\{a\}\times\{a\}]\} \tag{3.34}$$

となる．

$NGS=\phi$となったので，step2を終了し，step3を実行する．サブルーチン$select_ga$はOGS（式(3.34)）の中の各gaseに対し

$$select\_ga(sg(e,Y)?, sg[\{b\}\times\{b\}]) = \phi,$$

$$select\_ga(sg(e,Y)?, sg[\{e,f,g\}\times\{e,f,g\}]) = \{sg(e,e), sg(e,f), sg(e,g)\},$$

$$select\_ga(sg(e,Y)?, sg[\{c,d\}\times\{c,d\}]) = \phi,$$

$$select\_ga(sg(e,Y)?, sg[\{a\}\times\{a\}]) = \phi$$

を返すので，解集合

$$ANS = \{sg(e,e), sg(e,f), sg(e,g)\}$$

が得られる．

3.5 実験

直積問題クラス全体に適用できる従来の方法の中で，マジック集合法は最も効率的な方法の一つと思われる．直積法とマジック集合法の効率を比較するために計算機実験を行う．ルール集合と問い合わせアトムを固定した二つの直積問題の各々について，そのファクト集合をランダムに生成して多数の問題例を作り，これらの問題例に対して直積法とマジック集合法を適用して計算量を測定し，二つの方法の効率を比較する．

3.5.1 実験に使用する問題

実験に使用する二つの直積問題を説明する．

一つ目の問題はルール集合

$$P1: \quad s(X_1,X_2,X_3) \leftarrow A(X_1,X_2,X_3). \tag{3.35}$$

$$s(X_1,X_2,X_3) \leftarrow B_1(X_1',X_1), B_2(X_2',X_2), B_3(X_3',X_3), s(X_1',X_2',X_3'). \tag{3.36}$$

$$s(X_1,X_2,X_3) \leftarrow C_1(X_1',X_1), C_2(X_2',X_2), C_3(X_3',X_3), s(X_1',X_2',X_3'). \tag{3.37}$$

とファクト集合$D1$（ランダムに作成する）と問い合わせアトム$s(a_1,a_2,X_3)?$よりなる線形問題$(s(a_1,a_2,X_3)?,(P1,D1))$である．ここで，$A,B_1,B_2,B_3,C_1,C_2,C_3$はEDB述語，$s$はIDB述語，$a_1,a_2$は定数である．この問題を問題Aと呼ぶ．問題AはIDB述語の引数の数mが3の場合の同世代問題の再帰ルールを一つから二つに増やした問題であるが，再帰ルールが二つあるので，計数法等の同世代問題を対象とした方法では解くことはできない．問題Aは述語分割集合$\{s(X_1,X_2,X_3)\rightarrow s_1(X_1)\wedge s_2(X_2)\wedge s_3(X_3)\}$に関して直積問題になっている．

二つ目の問題はルール集合

$$P2: \quad s(X_1,X_2) \leftarrow E(X_1,X_2). \tag{3.38}$$

$$s(X_1,X_6) \leftarrow F_1(X_1,X_2), s(X_2,X_3), F_2(X_3,X_4), s(X_4,X_5), F_3(X_5,X_6). \tag{3.39}$$

とファクト集合$D2$（ランダムに作成する）と問い合わせアトム$s(a,X_2)?$よりなる非線形問題$(s(a,X_2)?,(P2,D2))$である．ここで，E,F_1,F_2,F_3はEDB述語，sはIDB述語，aは定数である．この問題を問題Bと呼ぶ．問題Bは$m=2$の場合の同世代問題の再帰ルールを非線形にした問題であるが，再帰ルールが非線形であるので，問題Bも，問題Aと同様，

計数法等の同世代問題を対象とした方法では解くことはできない．問題Bは，第3.2節で述べたように，述語分割集合$\{s(X_1, X_2) \rightarrow s_1(X_1) \wedge s_2(X_2)\}$に関して直積問題になっている．

3.5.2 計算量の測定の仕方

直積法とマジック集合法の計算量の測定の仕方を説明する．

初めに，直積法の時間量の測定の仕方を説明する．gase $p[C_1 \times \cdots \times C_h]$（直積$C_1 \times \cdots \times C_h$）における各集合の大きさの和$|C_1| + \cdots + |C_h|$を，そのgase（直積）の**長さ**と呼ぶ．直積法のstep1の時間量は初期化ルールよりgaseを生成するための時間量であり，これはstep1で生成されたgaseの長さの和（これをT_{CP_1}と記す）であると見なす．step2の主な時間量は，サブルーチン*gene_gase*が主ルールよりgaseを生成するための時間量と，被覆テストのための時間量と，サブルーチン*delete_gase*が不要なgaseを消去するための時間量である．gaseを生成するための時間量は生成型部分ルールを評価するための時間量と判定型部分ルールを評価するための時間量の和である．前者は，演繹データベースの研究でルールの評価時間を見積もるときに通常行われているように[BR86]，生成型部分ルールの成功回数の和であると見なす．また，後者は判定型部分ルールの成功回数の和であると見なすが，但し，判定型部分ルールはgase組1組に対し高々1回成功すると考える．gaseを生成するための以上の時間量，すなわち，生成型部分ルールと判定型部分ルールの成功回数の和をT_{CP_2g}と記す．被覆テストのための時間量は第3.3節の図3.3の各節点を生成するための時間量と，各節点において，それと共通部分をもつ直積を探すための時間量の和である．前者は節点（すなわち，直積）の長さの和（これをT_{CP_2c1}と記す）であると見なし，後者は識別番号集合IDの中の調べた要素の数の和（これをT_{CP_2c2}と記す）であると見なす．不要なgaseを消去するための時間量は，生成されたgase $p_0[C_{01} \times \cdots \times C_{0h}]$と$OGS$または$NGS$のgase（$p_0[C'_1 \times \cdots \times C'_h]$と記す）を比較して，$C'_1 \subseteq C_{01} \wedge \cdots \wedge C'_h \subseteq C_{0h}$を判定するための時間量であり，これは，集合$C'_i$の中の要素$c$が集合$C_{0i}$には含まれない（$c \in C'_i, c \notin C_{0i}$）ような集合$C'_i$および要素$c$を少なくとも1組見つけるまでに調べた，集合$C'_1, \ldots, C'_i$の中の要素の数の和（これを$T_{CP_2d}$と記す）であると見なす．step3の時間量はサブルーチン*select_ga*がgaseより解であるgaを作成するための時間量であり，これは，生成型部分ルールの評価時間の場合と同様，解ルールの成功回数の和（これをT_{CP_3}と記す）であると見なす．以上より，実験では，直積法の時間量として

$$T_{CP} = T_{CP\_1} + T_{CP\_2g} + T_{CP\_2c1} + T_{CP\_2c2} + T_{CP\_2d} + T_{CP\_3}$$

を計測する．

直積法の領域量の測定の仕方を説明する．直積法の主な領域量は，OGSおよびNGSの中のgaseを記憶するための領域量と，被覆テストにおいて一時的に作成される作業用直積を記憶するための領域量と，識別番号集合IDを記憶するための領域量と，ポインタ集合PTRを記憶するための領域量である．gase（作業用直積）のための領域量は記憶されている各gase（作業用直積）の長さの和であると見なす．集合ID（PTR）のための領域量は記憶されている各集合ID（PTR）の要素の数の和であると見なす．IDおよびPTRのための領域量は，各々，gaseのための領域量に等しい．これらの各領域量はアルゴリズムの進行と共に増減するが，各領域量の最大値の和を直積法の領域量と見なし，実験ではこれを計測する．

次に，マジック集合法の時間量および領域量の測定の仕方を説明する．マジック集合法は，初め，与えられた問題$(q(\boldsymbol{a}, \boldsymbol{X})?, (P, D))$のルール集合$P$を書き換えて新しいルール集合$P^{mg}$を作り，次に，$P$を$P^{mg}$に置き換えて得られる問題$(q(\boldsymbol{a}, \boldsymbol{X})?, (P^{mg}, D))$をセミナイーブ法を使って解くことにより，問題の解集合ANSを得る．マジック集合法の時間量はセミナイーブ法が問題$(q(\boldsymbol{a}, \boldsymbol{X})?, (P^{mg}, D))$を解くための時間量であり，これは，生成型部分ルールや解ルールの評価時間の場合と同様，すなわち，演繹データベースの研究において通常行われているように，P^{mg}の各ルールの成功回数の和であると見なす．また，マジック集合法の領域量は，セミナイーブ法が問題$(q(\boldsymbol{a}, \boldsymbol{X})?, (P^{mg}, D))$を解くための領域量であり，これはgaを記憶するための領域量，すなわち，「(記憶するgaの数) × (gaの引数の数)」であると見なす（本章ではマジック集合法の領域量として，gaの引数の数も考慮する）．

問題A，問題Bのどちらに対しても，マジック集合法の一つである一般化補助マジック集合法が効率的である．

一般化補助マジック集合法が問題Aに対して作るルール集合$P1^{mg}$を次に示す．

$P1^{mg}$:

$$m\_s(a_1, a_2) \leftarrow .$$
$$m\_sup_1(X'_1, X_2) \leftarrow m\_s(X_1, X_2), B_1(X'_1, X_1).$$
$$m\_s(X'_1, X'_2) \leftarrow m\_sup_1(X'_1, X_2), B_2(X'_2, X_2).$$
$$m\_sup_2(X'_1, X_2) \leftarrow m\_s(X_1, X_2), C_1(X'_1, X_1).$$
$$m\_s(X'_1, X'_2) \leftarrow m\_sup_2(X'_1, X_2), C_2(X'_2, X_2).$$
$$s(X_1, X_2, X_3) \leftarrow m\_s(X_1, X_2), A(X_1, X_2, X_3).$$

$sup_1(X_1, X_2, X_3') \leftarrow m\_s(X_1, X_2), B_1(X_1', X_1), B_2(X_2', X_2), s(X_1', X_2', X_3').$

$s(X_1, X_2, X_3) \leftarrow B_3(X_3', X_3), sup_1(X_1, X_2, X_3').$

$sup_2(X_1, X_2, X_3') \leftarrow m\_s(X_1, X_2), C_1(X_1', X_1), C_2(X_2, X_2'), s(X_1', X_2', X_3').$

$s(X_1, X_2, X_3) \leftarrow C_3(X_3', X_3), sup_2(X_1, X_2, X_3').$

同じく，一般化補助マジック集合法が問題Bに対して作るルール集合$P2^{mg}$を次に示す．

$P2^{mg}$: $m\_s(a) \leftarrow .$

$sup_1(X_1, X_2) \leftarrow m\_s(X_1), F_1(X_1, X_2).$

$sup_2(X_1, X_3) \leftarrow sup_1(X_1, X_2), s(X_2, X_3).$

$sup_3(X_1, X_4) \leftarrow sup_2(X_1, X_3), F_2(X_3, X_4).$

$sup_4(X_1, X_5) \leftarrow sup_3(X_1, X_4), s(X_4, X_5).$

$m\_s(X_2) \leftarrow sup_1(X_1, X_2).$

$m\_s(X_4) \leftarrow sup_3(X_1, X_4).$

$s(X_1, X_2) \leftarrow m\_s(X_1), E(X_1, X_2).$

$s(X_1, X_6) \leftarrow sup_4(X_1, X_5), F_3(X_5, X_6).$

3.5.3 実験結果

問題A，問題Bの各々に対し二つの実験を行った．初めに，問題Aに対する二つ実験（実験A1，A2と呼ぶ）について説明し，その後，問題Bに対する二つ実験（実験B1，B2と呼ぶ）について説明する．

問題Aの問題例の作り方を説明する．問題例のファクト集合$\{A$ファクト$\} \cup \{B_1$ファクト$\} \cup \cdots \cup \{C_3$ファクト$\}$は，異なる定数の数$n$とファクト密度を表すパラメタ$d$の二つのパラメータにしたがって，次のように作る．$A$ファクト集合は$\{A(i,i,i) \mid i = 1, \ldots, n\}$とする．$B_1$ファクト集合は$nd$個のファクトよりなる集合とし，各$B_1(i,j)$ファクトは$\{1, \ldots, n\}$の中からランダムに$i, j$を選ぶことにより作る．$B_2, B_3, C_1, C_2, C_3$ファクト集合も$B_1$ファクト集合と同様に作る．問題例の問い合わせアトムは常に$s(1, 1, X_3)$?とする．

実験A1では，nを$n = 50$に固定し，ファクト密度dを$d = 0.25, 0.5, 0.75, \ldots, 4.75, 5$と変化させ，そして，各$d$に対し$k = 5$個の問題例を作り，合計$1 \times 20 \times 5 = 100$個の問題例を作った．そして，これらの問題例に直積法とマジック集合法を適用した．実験結果を図

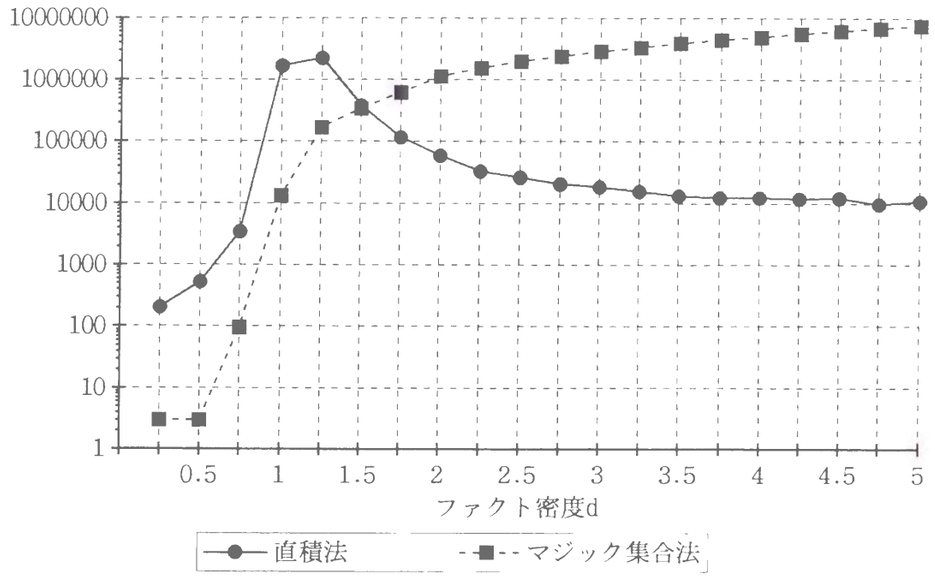


図3.4: 実験A1における平均時間量

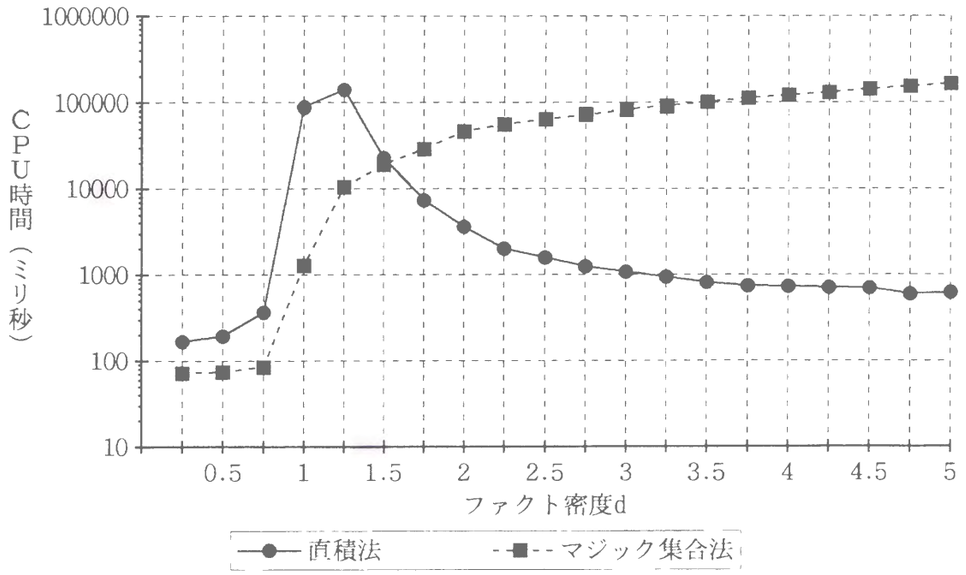


図3.5: 実験A1における平均CPU時間

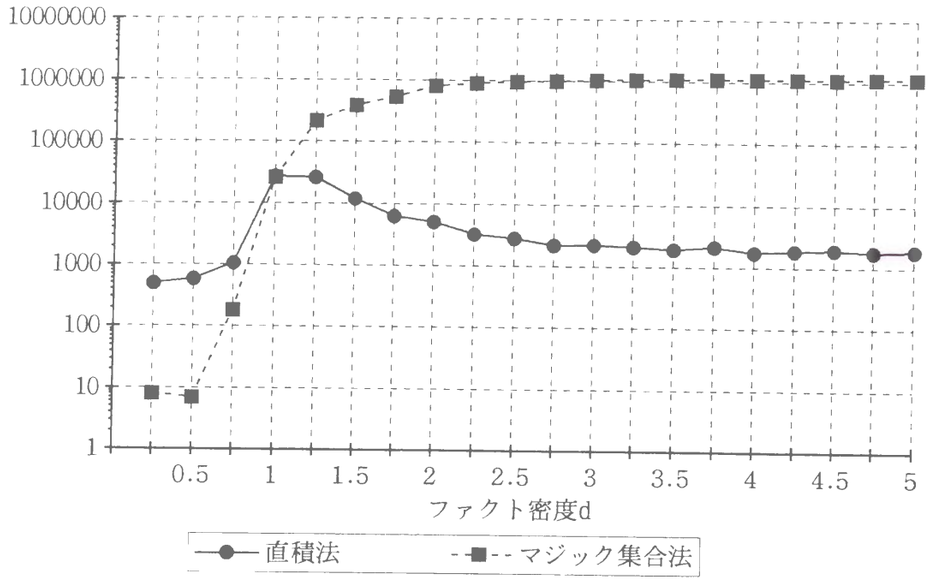


図3.6: 実験A1における平均領域量

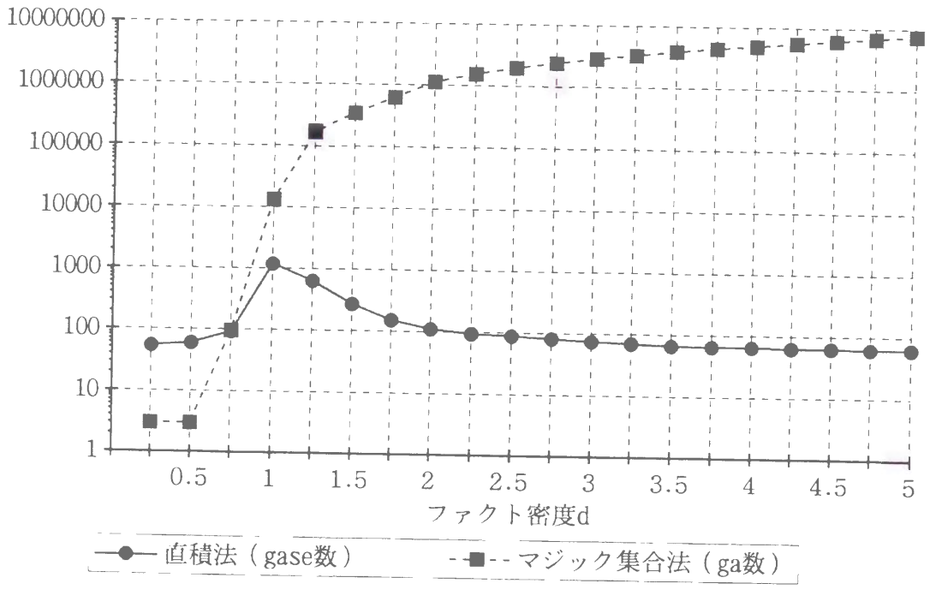


図3.7: 実験A1における平均gase数および平均ga数

3.4から図3.7に示す．これらのグラフの横軸はいずれもファクト密度d，縦軸は$k=5$個の問題例の平均値であり，そして，縦軸は対数目盛りになっている．図3.4は第3.5.2節で説明した時間量の平均値である．直積法の平均時間量はファクト密度dが1.25付近で爆発しているが，dが大きい所では小さくなっている．一方，マジック集合法の平均時間量はdが大きくなるにつれて増加している．ファクト密度dが大きい（$d \geq 1.5$）所では，直積法の平均時間量はマジック集合法のそれより小さい．図3.5は平均CPU時間である．図3.4の平均時間量の変化の仕方と図3.5の平均CPU時間の変化の仕方はほぼ等しいので，図3.4の時間量が各方法の時間量をほぼ正確に表していることが確認できる．図3.6は第3.5.2節で説明した領域量の平均値である．直積法の平均領域量は$d=1$付近で少し大きくなっているが，他の所では小さい．一方，マジック集合法の平均領域量はdが大きくなるにつれて増加し，$d \geq 2.5$ではほぼ一定となっている．ファクト密度dが大きい（$d \geq 1$）所では，直積法の平均領域量はマジック集合法のそれより小さい．図3.7は，直積法が生成したgaseの数の平均値，および，マジック集合法が生成したgaの数（重複して生成したgaの数も含む）の平均値である．マジック集合法が生成したgaの数はマジック集合法の時間量（図3.4）に等しいが，直積法のgaseの数と比較するために記した．直積法の生成gase数は$d=1$付近で少し大きくなっているが，他の所では小さい．ファクト密度が大きい（$d \geq 1$）所では，直積法の生成gase数はマジック集合法の生成ga数より小さい．

実験A1において，直積法の計算量が図3.4から図3.7のように変化した理由を説明する．

直積法の生成gase数が，ファクト密度dが中程度（$d=1$）の所で大きく，dが大きい所で小さいのは次の理由による．(1) dが小さいときは，主ルールからgaseは殆ど生成されない（但し，初期化ルールより50個のgaseが生成される）．(2) dが大きくなるにつれ，主ルールから小さなgaseが数多く生成されるようになる．(3) 更に，dが大きくなる（$d>1$）と，直積法の実行の早い時期に，大きなgaseが生成されるようになる．大きなgaseが幾つか生成されると，(3a) その後生成されるgaseの多くは，これらの大きなgaseに被覆されるので，NGSに保存されることなく捨てられる．また，(3b) NGSの中に保存されていた小さなgase（例えば初期化ルールより生成されたgase）の多くは，大きなgaseに包含されるために，NGSの中から消去される．NGSの中には，大きなgaseを主に，幾つかのgaseが残るが，(3c) それらのgaseも，gaseの生成に使用された後で，NGSからOGSへ移される．(3a)，(3b)，(3c)の結果，大きなgaseが幾つか生成された後は，NGSが急速に小さくなり，故に，gaseが生成されなくなる．このように，dが更に大きくなる（$d>1$）と，直積法の

実行の早い時期に大きなgaseを含む少数のgaseだけが生成されるようになる．(1),(2),(3)のために，直積法の生成gase数はdが中程度の所で大きく，dが大きい所で小さくなる．

直積法の平均領域量が，ファクト密度dが中程度（$d=1$）の所で大きく，dが大きい所で小さくなるのは，上に説明したように，生成gase数がそのように変化するからである．

直積法の平均時間量が，ファクト密度dが中程度（$d=1$）の所で爆発し，dが大きい所で小さくなるのは次の理由による．直積法の時間量のうち最も大きいのは被覆テストにかかる時間量である．被覆テストの全時間量は，被覆テスト1回当たりの時間量と被覆テストの回数の積である．被覆テスト1回当たりの時間量について考える．dが中程度の場合，上に述べたように，$NGS \cup OGS$には小さなgaseが数多く保存される．第3.3節の被覆テストの式(3.16)，すなわち，

$$C_{01} \times \cdots \times C_{0h} - C_{01}^1 \times \cdots \times C_{0h}^1 - C_{01}^2 \times \cdots \times C_{0h}^2 - \cdots - C_{01}^f \times \cdots \times C_{0h}^f$$

に現れる直積$C_{01}^1 \times \cdots \times C_{0h}^1, \ldots, C_{01}^f \times \cdots \times C_{0h}^f$はこのような多くの小さなgaseに対応している．すなわち，$C_{01}^1 \times \cdots \times C_{0h}^1, \ldots, C_{01}^f \times \cdots \times C_{0h}^f$において，各直積は小さく，かつ，直積の数$f$は多い．そのため，式(3.16)の計算過程を表す図3.3において，数多くの作業用直積が作成され，故に，被覆テスト1回当たりの時間量は大きくなる．一方，dが大きい場合，上に述べたように，$NGS \cup OGS$には大きなgaseが少しだけ保存される．すなわち，$C_{01}^1 \times \cdots \times C_{0h}^1, \ldots, C_{01}^f \times \cdots \times C_{0h}^f$において，各直積は大きく，かつ，直積の数$f$は少ない．そのため，図3.3において，少しの作業用直積しか作成されず，故に，被覆テスト1回当たりの時間量は小さくなる．このように，被覆テスト1回当たりの時間量は，dが中程度の所で大きく，dが大きい所では小さくなる．次に，被覆テストの回数について考える．被覆テストの回数は生成されるgaseの数に等しいので，先に述べたように，dが中程度の所で大きく，dが大きい所では小さくなる．以上，被覆テスト1回当たりの時間量と被覆テストの回数の両方が，dが中程度の所で大きく，dが大きい所では小さくなるために，被覆テストの全時間量は，dが中程度の所で爆発し，dが大きい所では小さくなり，その結果，直積法の平均時間量もそのように変化する．

実験A2について説明する．実験A2では，実験A1より規模の大きな（すなわち，nの大きな）問題例を作って，直積法の効率を調べた（マジック集合法はCPU時間およびメモリ量が爆発するために実験できない）．nを$n=50,100,\ldots,500$，dを$d=0.5,1,\ldots,5$，kを$k=5$として，合計$10 \times 10 \times 5 = 500$個の問題例を作り，これらの問題例に対し，タイムアウト時間を360000ミリ秒（すなわち，1時間）として，直積法を適用した．実験の結果，直積法の計算量は実験A1のときとほぼ同様な変化の仕方をし，ファクト密度dが大きい場合，直積法はnの大きな問題例も効率的に解くことができた．例えば，$n=500$の場合，$d=2$の問題例はタイムアウトしたが，$d=2.5$の問題例は平均時間量688594，平均CPU時間43980ミリ秒で，また，$d=5$の問題例は平均時間量179321，平均CPU時間10774ミリ秒で解くことができた．

以上の実験A1およびA2より，問題Aに対し，ファクト密度が高いとき，直積法がマジック集合法より効率的であることが分かる．

次に，実験B1およびB2について説明する．実験B1およびB2では，問題Bに対し，実験A1およびA2と同様の実験を行った．

問題Bの問題例の作り方を説明する．問題例のファクト集合$\{E\text{ファクト}\} \cup \{F_1\text{ファクト}\} \cup \{F_2\text{ファクト}\} \cup \{F_3\text{ファクト}\}$は，異なる定数の数$n$とファクト密度$d$の二つのパラメータにしたがって，次のように作る．$E$ファクト集合は$nd$個のファクトよりなる集合とし，各$E(i,j)$ファクトは$\{1,\ldots,n\}$の中からランダムに$i,j$を選ぶことにより作る．$F_1,F_2,F_3$ファクト集合も$E$ファクト集合と同様に作る．各問題例の問い合わせアトムは常に$s(1,X_2)?$とする．

実験B1では，nは$n=100$とし，また，dとkは実験A1のそれらと同じとして，合計$1 \times 20 \times 5 = 100$個の問題例を作り，これらの問題例に直積法とマジック集合法を適用した．平均時間量を図3.8に示す．直積法の平均時間量は，ファクト密度dが$d=1.25$付近で爆発しているが，dが大きい所では小さくなっている．マジック集合法の平均計算量はdが大きくなるにつれて増加し，$d \geq 4$ではほぼ一定になっている．平均領域量を図3.9に示す．直積法の平均領域量は，$d=1.25$付近で少し大きくなっているが，他の所では小さい．マジック集合法の平均領域量は，dの増加とともに大きくなり，$d \geq 4$ではほぼ一定となっている．平均時間量と平均領域量のどちらの計算量においても，実験A1のときと同様，ファクト密度dが大きい所では，直積法の計算量はマジック集合法のそれより小さい．

実験B2では，実験B1より規模の大きな（すなわち，nの大きな）問題例を作り，直積法の効率を調べた（実験A2のときと同様，マジック集合法はCPU時間およびメモリ量が爆発するために実験できない）．nを$n=100,200,\ldots,1000$とし，dとkは実験A2のそれらとして，合計$10 \times 10 \times 5 = 500$個の問題例を作り，これらの問題例に対し，タイムアウト時間を360000ミリ秒（すなわち，1時間）として，直積法を適用した．実験の結果，直積法の計

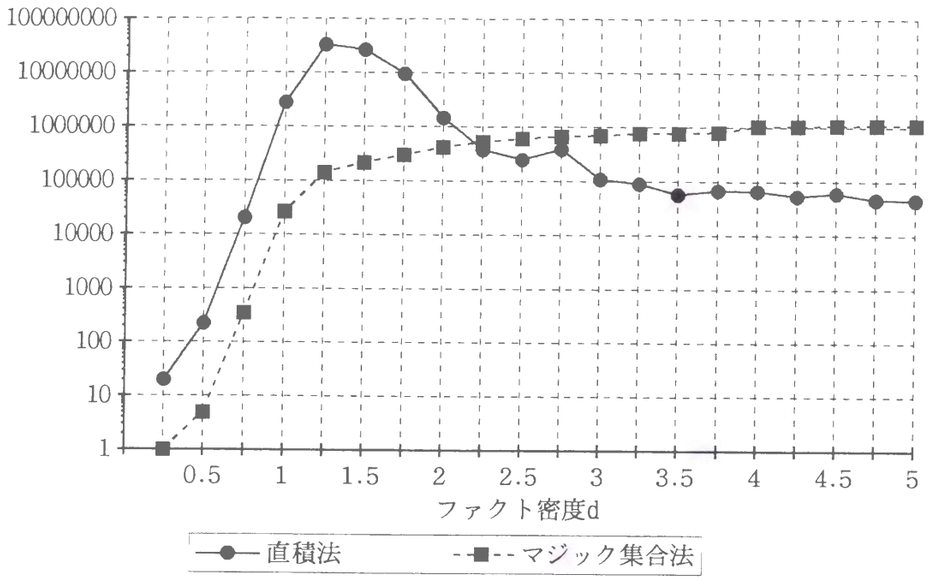


図3.8: 実験B1における平均時間量

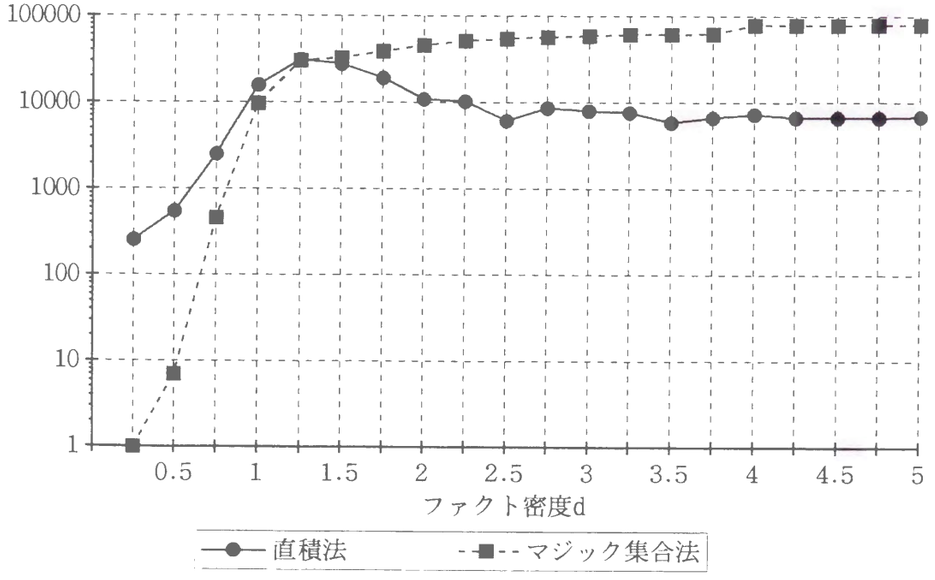


図3.9: 実験B1における平均領域量

算量は実験B1のときとほぼ同様な変化の仕方をし，ファクト密度dが大きい場合，直積法はnの大きな問題例も効率的に解くことができた．例えば，$n = 1000$の場合，$d = 3.5$の問題例はタイムアウトしたが，$d = 4$の問題例は平均時間量3290968，平均CPU時間153000ミリ秒で，また，$d = 5$の問題例は平均時間量1893745，平均CPU時間82556ミリ秒で解くことができた．

以上の実験B1およびB2より，問題Bに対しても，問題Aと同様，ファクト密度が高いとき，直積法がマジック集合法より効率的であることが分かる．

3.6 むすび

直積問題クラスと呼ぶdatalog問題クラスの新しい部分クラスを定義し，この問題クラスを効率的に解くための方法として直積法を提案した．直積法の効率と，直積問題クラス全体に適用できる従来の方法の中で最も効率的なマジック集合法の効率を比較するために，直積問題の例として，同世代問題の再帰ルールを複数にした問題，および，非線形にした問題を考え，これらの問題を使って計算機実験を行った．実験の結果，どちらの問題に対しても，ファクト集合がファクトを密に含む場合，直積法がマジック集合法より効率的であることを示した．

第4章

多次元直方体被覆問題を効率的に解くためのアルゴリズム

本章では，直積問題を制限した連続直積問題に直積法を適用したときに直積法の内部処理に現れる組合せ問題で，この組合せ問題を効率的に解ければ，その解法を直積法に組み込むことにより，連続直積問題に対する直積法の効率を改善できる組合せ問題について考える．この組合せ問題は，一つのm次元直方体（全領域と呼ぶ）が複数のm次元直方体の和により被覆されているか否かを判定する問題であり，多次元直方体被覆問題と呼ぶ．

多次元直方体被覆問題の研究はこれまであまり見受けられないが，この問題は充足可能性問題の素直な一般化になっているので，充足可能性問題の代表的解法であるDavis-Putnum法と局所探索法を素直に拡張して，この問題を解くことができる．

多次元直方体被覆問題を効率的に解くための新しい解法として，直方体群の幾何学的性質を利用してDavis-Putnum法を拡張した新しい解法（変形Davis-Putnum法と呼ぶ）を提案する．提案した解法および従来の解法の効率を調べるために計算機実験を行い，その結果，(1)提案した変形Davis-Putnum法が，全領域の全体または全領域のほとんどの部分が被覆されていて，かつ，直方体の重複の度合が低い問題例に対し，従来のDavis-Putnum法および局所探索法より効率的であること，および，(2)これら三つの解法を併用すれば，様々な性質をもった問題例の多くを効率的に解けること，を示す．

先に述べた直積法と多次元直方体被覆問題の関係，および，(2)の結果として，変形Davis-Putnum法，Davis-Putnum法，局所探索法の三つの解法を直積法に組み込むことにより，連続直積問題の多くに対して直積法をより効率化することができる．

4.1 はじめに

第3章の直積問題にファクト集合が“連続である”等の制限を加えた問題を**連続直積問題**（continuous Cartesian product problem）と呼ぶ．本章では，連続直積問題に第3章の直積法を適用した場合に直積法の内部処理に現れる組合せ問題で，この組合せ問題を効率的に解ければ，その解法を直積法に組み込むことにより，連続直積問題に対する直積法の効率を改善できる組合せ問題について考える．この組合せ問題は，一つのm次元直方体が複数のm次元直方体の和により被覆されているか否かを判定する問題であり，**多次元直方体被覆問題**（multi-dimensional rectangle cover problem）と呼ぶ（直積法と多次元直方体被覆問題の関係については，連続直積問題の定義も含めて，第4.2節で詳しく説明する）．

多次元直方体被覆問題は，具体的には，次のように表される．**m次元直方体**(m-dimensional rectangle)は，正整数mと$b_i < e_i, i = 1, \ldots, m$, であるある整数$b_1, \ldots, b_m, e_1, \ldots, e_m$を使って表される$m$次元ユークリッド空間内の閉領域$\{(x_1, \ldots, x_m) \mid b_i \leq x_i \leq e_i,\ x_i$は実数，$i = 1, \ldots, m\}$であり，$R(b_1, \ldots, b_m, e_1, \ldots, e_m)$または，単に，$R, R', R_1, \ldots$などと記す．**多次元直方体被覆問題**は，$m$次元直方体$R_0$（**全領域**（full domain）と呼ぶ）とR_0に包含された複数のm次元直方体$R_1, \ldots, R_n$が与えられて，R_0が$R_1, \ldots, R_n$の和により被覆されている（すなわち，$R_0 = R_1 \cup \cdots \cup R_n$）か否かを判定する（正確には，“被覆されていないか?”に答える）問題であり，$(R_0, \{R_1, \ldots, R_n\})$と記す．例えば，図4.8(a)は$m = 2, n = 4$の場合の問題例であり，この例は被覆されていない．

多次元直方体被覆問題は，NP完全で有名な**充足可能性問題**（SAT問題と記す）の素直な一般化になっている．以下に，SAT問題が，全領域R_0が$R_0 = R(0, \ldots, 0, 2, \ldots, 2)$に限定された多次元直方体被覆問題として表せることを，例を使って説明する．3変数よりなるCNF論理式$E = (\overline{x}+\overline{y})(x+y+\overline{z})$を考える．$E$の否定$\overline{E} = xy+\overline{x}\overline{y}z$の各項に対し，それを真とする変数値$(x, y, z)$の集合を$\{0,1\}^3$空間上にとるとき，$E$の充足可能性を判定する問題は，$\{0,1\}^3$空間がこのように作られた点集合の和$\{1\}\times\{1\}\times\{0,1\}\cup\{0\}\times\{0\}\times\{1\}$により被覆されているか否かを判定する問題となる[Iwa89]．すなわち，Eが充足可能（不能）のとき，$\{0,1\}^3$空間は被覆されない（される）．更に，この$\{0,1\}^3$空間上の被覆問題は，$\{0,1\}^3$空間の各点(a, b, c)を3次元ユークリッド空間の領域$\{(x, y, z) \mid a \leq x \leq a+1, b \leq y \leq b+1, c \leq z \leq c+1\}$に対応させるとき，$R_0 = R(0, 0, 0, 2, 2, 2), R_1 = R(1, 1, 0, 2, 2, 2), R_2 = R(0, 0, 1, 1, 1, 2)$の多次元直方体被覆問題となる．このように，SAT問題は$R_0 = R(0, \ldots, 0, 2, \ldots, 2)$の多次

元直方体被覆問題として表すことができる．多次元直方体被覆問題はNP完全問題を表すことができ，かつ，クラスNPに属すので，ＮＰ完全である．

従来，多次元直方体被覆問題そのものの研究はあまり見受けられないが，SAT問題については多くの研究が行われている．SAT問題を効率的に解くためのさまざまな解法が提案され，幾つかの解法については平均計算量が理論的解析または計算機実験により評価されている[BP81, GPB82, PB85, Pur87, Pur90, Iwa89, Fra91, SLM92, MI93, OYO93]．それらの中で，**Davis-Putnum法**(DP法と略す)[DLL62, GPB82]と**局所探索法**(local search法，LS法と略す)[SLM92, MI93]は代表的な解法である．文献[BP81, GPB82, PB85, Pur87, Pur90]には，簡略化したDP法をランダムSAT問題に適用した場合の平均計算量が理論的に解析されている．これらの文献によると，例えば，変数の数mとリテラルが節に現れる確率p'を固定し，節の数nを小さな値から大きな値に変化させて問題例を作った場合，nが小さい場合と大きい場合はDP法は効率的であるが，その中間のnに対してはDP法の効率は低下する．また，文献[SLM92]には，ランダム3SAT問題に対するDP法とLS法の実験結果がある．実験の結果として，(i) ランダム3SAT問題では$n = 4.3m$程度の問題例がDP法にとって一番難しいこと，(ii) そのような問題例に対して，DP法は$m = 120, n = 516$程度の規模の問題例までは解くことができるが，それ以上の規模になると急に効率が低下すること，(iii) LS法は，そのような問題例の中の充足可能な問題例に対して，DP法より大幅に効率的であること，が述べられている．但し，DP法とLS法の本質的相違点として，DP法は任意の問題例を解くことができるが，LS法は充足不能な問題例を解くことができない．また，充足可能な問題例であっても，LS法がそれを解く保障はない．

先に述べたように，多次元直方体被覆問題はSAT問題の素直な一般化になっているので，SAT問題のためのDP法もLS法も，素直な拡張により，多次元直方体被覆問題に適用することができる．以後，多次元直方体被覆問題に拡張されたDP法，LS法も，単に，DP法，LS法と呼ぶ．

多次元直方体被覆問題を効率的に解くための新しい解法として，直方体群の幾何学的性質を利用してDavis-Putnum法を拡張した新しい解法を提案する．この解法を**変形Davis-Putnum法**と呼び，MDP法と略す．ある領域DMが直方体$R'_1, \ldots, R'_{n'} (\subseteq DM)$により被覆されているか否かを判定する場合を考える．MDP法では，DMが被覆されているか否かの状態を変更しないように保障しながら$R'_1, \ldots, R'_{n'}$の縮小，消去，併合を繰り返すことにより，直方体集合$\{R'_1, \ldots, R'_{n'}\}$を直方体数の少ない集合$\{R''_1, \ldots, R''_h\}, h \leq n'$,に変形する．変形によってすべての直方体が消去された$(h = 0)$場合には，DMが被覆されていなかったことが分かり，DMと等しい直方体$R''(= DM)$が生成された場合には，DMが被覆されていたことが分かる．これらのどちらでもない場合は，DMを領域DM_1, DM_2に分割し，DM_1, DM_2と変形後の直方体集合$\{R''_1, \ldots, R''_h\}$から問題$(DM_1, \{DM_1 \cap R''_1, \ldots, DM_1 \cap R''_h\})$と$(DM_2, \{DM_2 \cap R''_1, \ldots, DM_2 \cap R''_h\})$を作り，それらの各問題に対し上記の処理を繰り返す．MDP法はDP法に直方体の変形処理を加えたものである．

一番初めの問題を$(R_0, \{R_1, \ldots, R_n\})$として，これに上記の処理を繰り返し適用した結果生成される問題$(DM_1, \{DM_1 \cap R''_1, \ldots, DM_1 \cap R''_h\}), (DM_2, \{DM_2 \cap R''_1, \ldots, DM_2 \cap R''_h\})$などを$(R_0, \{R_1, \ldots, R_n\})$の**部分問題**（sub problem）と呼ぶ．また，便宜的に$(R_0, \{R_1, \ldots, R_n\})$自身も部分問題と呼ぶ．MDP法とDP法の効率を比較した場合，一つの部分問題に対する計算量は直方体の変形操作が加わっているだけMDP法の方がDP法より大きいが，部分問題の数がMDP法の方がDP法より少なくなれば，全計算量はMDP法の方がDP法より少なくなる可能性がある．

多次元直方体被覆問題に対するMDP法，DP法，LS法の効率を調べるために計算機実験を行う．ここで，実験に使用するLS法はSAT問題のための最も基本的なLS法を多次元直方体被覆問題用に拡張したものであり，先の文献[SLM92]のLS法を拡張したものではない（第4.3.2節）．また，問題例は三つの方法で作る．各方法で作る問題例を**分割型問題例**，**拡大型問題例**，**ランダム型問題例**と呼ぶ．ランダム型問題例は先の文献[BP81]等のランダムSAT問題例の作り方を多次元直方体被覆問題用に拡張した方法を使って作る．この問題例では，直方体の重複の度合いが高いと被覆され，低いと被覆されない．分割型問題例は新しい方法によって作る問題例であり，この問題例では，直方体の重複の度合いを一定に保ったまま，被覆された問題例から被覆されていない問題例まで幅広い問題例を作ることができる．但し，分割型問題例では，一部の問題例において直方体が整然と並ぶという性質がある．拡大型問題例は，この性質を軽減するために，分割型問題例の直方体を更に少しランダムに拡大して作る問題例である．

以上の3種類の問題例に対する実験により，(1) 提案したMDP法が，全領域の全体または全領域のほとんどの部分が被覆されていて，かつ，直方体の重複の度合が低い問題例に対し，従来のDP法およびLS法より効率的であること，および，(2) どの解法も，それ一つだけでは，様々な性質をもった問題例全体を効率的に解くことはできないが，しかし，これら三つの解法を併用すれば，そのような問題例の多くを効率的に解けること，を示す．

初めに述べた直積法と多次元直方体被覆問題の関係，および，(2)の結果として，MDP法，DP法，LS法の三つの解法を直積法に組み込むことにより，連続直積問題の多くに対して直積法をより効率化することができる．

本章の以降は次のように構成される．第4.2節で直積法と多次元直方体被覆問題の関係を詳しく説明する．第4.3節で多次元直方体被覆問題に拡張されたDP法とLS法を与え，そして，第4.4節でMDP法を与える．第4.5節で実験を示し，最後に第4.6節でまとめる．

4.2 直積法と多次元直方体被覆問題

準備として用語を幾つか定義した後，連続直積問題を定義する．次に，連続直積問題に第3章の直積法を適用したとき，直積法の内部処理に多次元直方体被覆問題が現れることを説明する．最後に，多次元直方体被覆問題を効率的に解くことができれば，連続直積問題に対して直積法をより効率化できることを説明する．

述語分割集合PD（第3.2節を参照）は，その中の各述語分割が，$p(X_1,\ldots,X_m) \rightarrow p_1(X_1) \wedge \cdots \wedge p_m(X_m)$のように，述語$p$を引数を一つだけもつ複数の述語$p_1,\ldots,p_m$に分割するとき，**原始的である**（primitive）と言う．

原始的な述語分割集合PDにしたがって，直積問題の主ルールを分割したときにできる各生成型部分ルール（第3.2節を参照）を

$$r: \quad p_0(X_0) \leftarrow A_1(\boldsymbol{Y}_1),\ldots,A_l(\boldsymbol{Y}_l),p_1(X_1),\ldots,p_h(X_h). \tag{4.1}$$

と記す．ここで，$X_0,X_1,\ldots,X_h$は変数，$\boldsymbol{Y}_1,\ldots,\boldsymbol{Y}_l$は変数ベクトルであり，$X_0,X_1,\ldots,X_h$の中には同じ変数が含まれてもよい．また，$A_1(\boldsymbol{Y}_1),\ldots,A_l(\boldsymbol{Y}_l)$はEDBアトム，$p_0(X_0)$, $p_1(X_1),\ldots,p_h(X_h)$は主ルールのIDBアトムを分割して得られるアトム（これもIDBアトムと呼ぶ）である．各EDBアトム$A_i(\boldsymbol{Y}_i), i=1,\ldots,l$,は引数に同じ変数を複数持たないと仮定する．すなわち，各EDBアトムは$A(X,X,Y)$のようではなく，$A(X,Y,Z)$のようであるとする．このとき，生成型部分ルールrより次のように無向グラフGを作る．Gの節点はrのアトムおよび変数である．Gの枝は，アトム$p_i(X_i)$（$A_j(\boldsymbol{Y}_j)$）が変数Xを含むとき，$p_i(X_i)$（$A_j(\boldsymbol{Y}_j)$）とXの間に枝を作る．このグラフGを生成型部分ルールrのための**アトム接続グラフ**（atom connection graph）と呼ぶ．例えば，生成型部分ルールrを

$$p(X) \leftarrow A(X,Y,Z),B(Y,V),C(Y),q(Z),s(V). \tag{4.2}$$

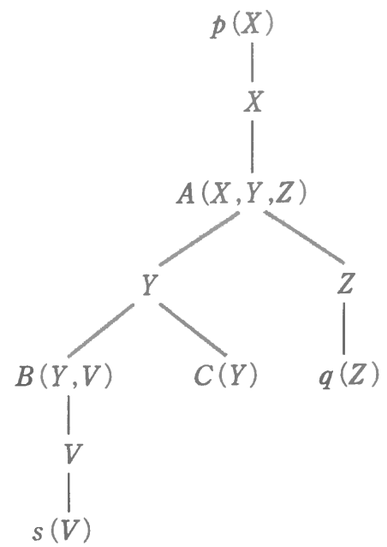


図4.1: アトム接続グラフ

とするとき，そのアトム接続グラフは図4.1になる．

ファクト集合Dの中のEDB述語Aのファクトの集合をD_A（$=\{A(a_1,\ldots,a_k) \mid A(a_1,\ldots,a_k) \in D\}$）と記す．$A$ファクト集合$D_A$は，次の条件1，2を満たすとき，**連続である**（continuous）と言う．

1. 定数はすべて整数である．
2. $A(a_1,\ldots,a_k)$と$A(b_1,\ldots,b_k)$をD_Aの中の任意のファクト，i（$i \in \{1,\ldots,k\}$）を任意の引数位置，c_iを$\min\{a_i,b_i\} \leq c_i \leq \max\{a_i,b_i\}$を満たす任意の整数とするとき，$\min\{a_j,b_j\} \leq c_j \leq \max\{a_j,b_j\}$（$j=1,\ldots,k, j \neq i$）を満たすあるファクト$A(c_1,\ldots,c_k)$が$D_A$の中に少なくとも一つ存在する．

例えば，Aファクト集合$D_A=\{A(1,1,2),A(1,2,3),A(2,2,4),A(3,2,4)\}$は連続である．

定義 4.1 述語分割集合PDに関する直積問題が次の条件を満たすとき，その直積問題は**連続直積問題**（continuous Cartesian product problem）であると言う．

1. PDは原始的である．
2. 主ルールに現れる各EDBアトムは引数に同じ変数を複数持たない．

3. 主ルールを分割して得られる各生成型部分ルールにおいて，そのアトム接続グラフは閉路を含まない．

4. 定数はすべて整数である．

5. 主ルールに現れる各EDB述語Aについて，ファクト集合Dの中のAファクトの集合D_Aは連続である．□

上のように定義した連続直積問題に直積法を適用した場合を考える．

補題 4.1 連続直積問題に直積法を適用した場合，生成されるgase $p[C_1 \times \cdots \times C_m]$の中の各集合$C_i, i = 1, \ldots, m,$は連続した整数の集合になる．すなわち，$C_i$は$a'_i \leq b'_i$であるある整数$a'_i$と$b'_i$を使って，$C_i = \{x \mid a'_i \leq x \leq b'_i,\ x$は整数$\}$と表すことができる．

証明 述語分割集合PDは原始的であり，かつ，定数はすべて整数であるので，gase $p[C_1 \times \cdots \times C_m]$の中の各集合$C_i, i = 1, \ldots, m,$は整数の集合となる．従って，補題を証明するためには，「整数集合C_iが連続となる（命題1と呼ぶ）」ことを示せばよい．

命題1を帰納法を使って証明することを考える．直積法が初期化ルールより生成するgaseについては，その中の集合C_iは，整数を一つだけ含むので，連続である．従って，帰納法の証明を完成させるためには，「主ルールの本体に代入する各gaseについて，その中の各集合Cが連続であるならば，この主ルールより生成されるgaseについても，その中の各集合C'が連続になる（命題2と呼ぶ）」ことを示せばよい．

命題2を証明するためには，「主ルールを分割してできる各生成型部分ルール（式(4.1)とする）から生成されるga集合$\{p_0(c) \mid c \in C'\}$において，集合C'が連続になる（命題3と呼ぶ）」ことを示せばよい．

以下，例を使って，命題3を証明する．生成型部分ルールを式(4.2)とする．ルール(4.2)は直積問題の条件2および3を満たしている．ルール(4.2)の本体のIDB述語q（s）のためのga集合を$\{q(c) \mid c \in C_q\}$（$\{s(c) \mid c \in C_s\}$）と記す．ここで，仮定より，C_q（C_s）は連続した整数の集合である．また，これらのga集合と連続ファクト集合D_A, D_B, D_Cとルール(4.2)を使って生成されるp gaの集合を$\{p(c) \mid c \in C_p\}$と記し，C_pの中の最小値をa_X，最大値をb_Xと記す．更に，ga $p(a_X)$を生成するためにルール(4.2)の本体に代入されたgaを$A(a_X, a_Y, a_Z), B(a_Y, a_V), C(a_Y), q(a_Z), s(a_V)$と記し，同様に，ga $p(b_X)$を生成するためにルール(4.2)の本体に代入されたgaを$A(b_X, b_Y, b_Z), B(b_Y, b_V), C(b_Y), q(b_Z), s(b_V)$と記す．このとき，命題3すなわち「$a_X \leq c_X \leq b_X$である任意の整数$c_X$に対し，ga $p_0(c_X)$がルール(4.2)より生成される（命題4と呼ぶ）」ことは，次のように証明することができる．ルール(4.2)のアトム接続グラフは，連続直積問題の条件3より閉路を含まないので，図4.1のように，$p(X)$を根とする木として表せる．アトム接続グラフの根から葉に向かってファクト集合およびga集合を調べていく．(i) Aファクト集合D_Aについて考える．D_Aは連続であるので，$\min\{a_Y, b_Y\} \leq c_Y \leq \max\{a_Y, b_Y\}, \min\{a_Z, b_Z\} \leq c_Z \leq \max\{a_Z, b_Z\}$であるようなあるファクト$A(c_X, c_Y, c_Z)$が$D_A$の中に存在する．(ii) Bファクト集合D_Bについて考える．$\min\{a_Y, b_Y\} \leq c_Y \leq \max\{a_Y, b_Y\}$であり，かつ，$D_B$は連続であるので，$\min\{a_V, b_V\} \leq c_V \leq \max\{a_V, b_V\}$であるようなあるファクト$B(c_Y, c_V)$が$D_B$の中に存在する．(iii) ga 集合$\{c \mid c \in C_s\}$について考える．$\min\{a_V, b_V\} \leq c_V \leq \max\{a_V, b_V\}$であり，かつ，集合$C_s$は連続であるので，ga $s(c_V)$はga 集合$\{s(c) \mid c \in C_s\}$の中に存在する．同様に議論を進めると，ファクト$C(c_Y)$がD_Cの中に存在すること，および，ga $q(c_Z)$がga 集合$\{q(c) \mid c \in C_q\}$の中に存在することが分かる．以上，ga $p_0(c_X)$を生成する$A(c_X, c_Y, c_Z), B(c_Y, c_V), C(c_Y), s(c_V), q(c_Z)$がファクト集合およびga集合の中に存在するので，ルール(4.2)を評価する際には，それらがルール(4.2)の本体に代入されて，ga $p(c_X)$が生成される．これにより，命題4が証明された．従って，補題が証明された．□

上の補題より，連続直積問題に直積法を適用した場合，生成されるgase $p[C_1 \times \cdots \times C_m]$の中の各集合$C_i, i = 1, \ldots, m,$は連続した整数の集合となる．従って，直積法が被覆テストにおいて判定する第3章の式(3.15):

$$C_{01} \times \cdots \times C_{0h} \not\subseteq \bigcup_{p_0[C'_{01} \times \cdots \times C'_{0h}] \in NGS \cup OGS} C'_{01} \times \cdots \times C'_{0h}$$

の中の各集合C_{0i}, C'_{0j}も連続した整数の集合となる．故に，式(3.15)を判定する問題は，各直積を多次元直方体と考えて，多次元直方体被覆問題と見なすことができる．

多次元直方体被覆問題は，連続性を利用すれば，C_{0i}, C'_{0j}が連続でない場合の式(3.15)の判定問題より効率的に解ける可能性がある．もし，多次元直方体被覆問題を効率的に解くことができれば，その解法を第3章の直積法に組み込むことにより，被覆テストをより効率化することができる．直積法の計算時間のほとんどは被覆テストを繰り返し行うための時間であったので，故に，連続直積問題に対しては，直積法の効率を更に高めることができる．

step1: $ST := \{R_0\}$;
step2: $ST = \emptyset$のとき，R_0は被覆されているとして終了;
step3: STから領域を一つ（DMと記す）取り出し，
（ケース1）$DM \subseteq R_i$である直方体$R_i, i \in \{1, \ldots, n\}$, が存在する場合，step2へ;
（ケース2）$DM \cap R_i$が体積をもつ直方体R_iが存在する場合，単一節分割または純リテラル分割または単純分割によりDMを領域DM', DM'''に分割してそれらをSTに加え，step2へ;
（ケース3）その他の場合，R_0は被覆されていないとして終了;

図4.2: 多次元直方体被覆問題に拡張されたDP法

4.3 従来の解法

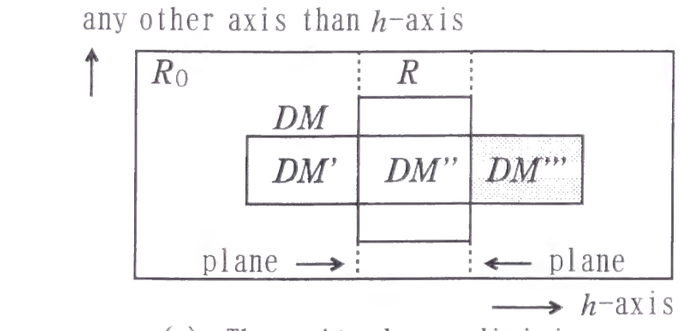
4.3.1 多次元直方体被覆問題のためのDP法

本章では，3次元ユークリッド空間における平面，体積，面積などの用語をm次元ユークリッド空間においても使用する．m次元直方体$R(b_1, \ldots, b_m, e_1, \ldots, e_m)$が与えられたとき，$(m-1)$次元領域$\{(x_1, \ldots, x_m) \mid x_h = b_h(\text{or } e_h),\ -\infty < x_i < +\infty (i \neq h)\}, h = 1, \ldots, m,$を$R$の**境界平面**（boundary plane）と呼び，$x_h = b_h(\text{or } e_h)$と記す．また，$R$の各境界平面と$R$との共通部分（例えば$\{(x_1, \ldots, x_m) \mid x_h = b_h,\ b_i \leq x_i \leq e_i (i \neq h)\}$）を$R$の**周囲面**（peripheral plane）と呼ぶ．m次元空間内の領域$V = \{(x_1, \ldots, x_m) \mid b'_i \leq x_i \leq e'_i,\ i = 1, \ldots, m\}$を考える．$b'_i < e'_i, i = 1, \ldots, m,$の場合，$V$は**体積をもつ**と言い，$(e'_1 - b'_1) \cdots (e'_m - b'_m)$を$V$の**体積**（volume）と呼び，$|V|$と記す．また，$b'_h = e'_h, b'_i < e'_i (i \neq h)$の場合，$V$は**面積をもつ**と言う．

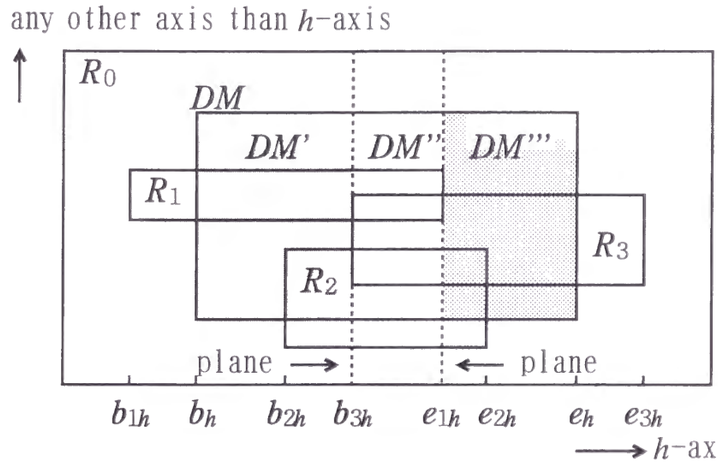
多次元直方体被覆問題に拡張されたDP法を図4.2に示す．STはR_0の分割によりできた領域を入れるスタックである．step3の**単一節分割**（unit clause division），**純リテラル分割**（pure literal division）および**単純分割**（simple division）はSAT問題に対するDP法[GPB82]の分割法を拡張して用いている．

単一節分割では，図4.3(a)のように，領域DMを直方体Rの平行な境界平面により三つの領域DM', DM'', DM'''に分割したときに，RがDM''を包含するような直方体Rが存在する場合，DMをそのように分割し，DM'とDM'''のみをスタックSTに加える．

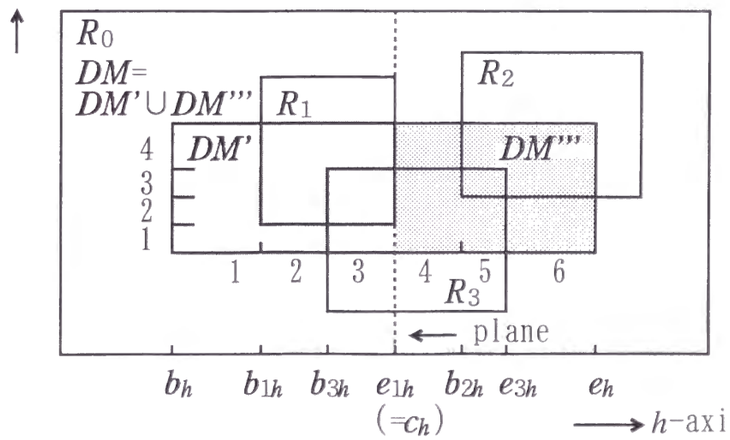
純リテラル分割では，図4.3(b)のように，DMとの共通部分が体積をもつすべての直方体



(a) The unit clause division



(b) The pure literal division



(c) The simple division

図4.3: 領域の分割方法

をR_1, R_2, R_3として，DM, R_1, R_2, R_3の各h座標区間$[b_h, e_h], [b_{1h}, e_{1h}], [b_{2h}, e_{2h}], [b_{3h}, e_{3h}]$が共通区間をもつ（すなわち$\max\{b_h, b_{1h}, b_{2h}, b_{3h}\} < \min\{e_h, e_{1h}, e_{2h}, e_{3h}\}$）ような座標軸$h$が存在する場合，平面$x_h = \max\{b_h, b_{1h}, b_{2h}, b_{3h}\}$と$x_h = \min\{e_h, e_{1h}, e_{2h}, e_{3h}\}$により$DM$を三つの領域$DM', DM'', DM'''$に分割し，$DM'$と$DM'''$のみをスタック$ST$に加える．$DM''$を$ST$に加えないのは，$DM' \cup DM'''$が被覆されていれば$DM''$も被覆されるため，$DM$の被覆状態を$DM' \cup DM'''$の被覆状態のみで決定できるからである．

単純分割では，図4.3(c)のように，DMの区間の中で“長さ”の最も“長い”区間をh座標区間$[b_h, e_h]$とし，その区間の“中点”をc_hとするとき，平面$x_h = c_h$によりDMを2分割し，できた領域DM', DM'''をスタックSTに加える．但し，ここに限り，各区間の“長さ”，例えばh座標区間の“長さ”とは，DMとの共通部分が体積をもつ直方体（R_1, R_2, R_3とする）のh軸に垂直な境界平面の中で，DMと交わるもの（$x_h = b_{1h}, x_h = b_{3h}, x_h = e_{1h}, x_h = b_{2h}, x_h = e_{3h}$とする）の数$+1$（すなわち，6）と定義し，$h$座標区間の“中点”$c_h$とは，これらの境界平面$x_h = b_{1h}, x_h = b_{3h}, x_h = e_{1h}, x_h = b_{2h}, x_h = e_{3h}$の$h$座標$b_{1h}, b_{3h}, e_{1h}, b_{2h}, e_{3h}$の中央値（すなわち，$e_{1h}$）と定義する．

なお，分割の優先順序は，SAT問題に対するDP法[GPB82]のそれにしたがって，単一節分割，純リテラル分割，単純分割の順とする．

4.3.2 多次元直方体被覆問題のための局所探索法

図4.4の$m = 2, n = 4$の問題例を使って用語を説明する．この図で，R_0は直方体R_1, R_2, R_3, R_4の境界平面により$(2n+1)^m = 81$個の領域に分割される．これらの各領域を**セル**（cell）と呼ぶ．一つのセルCから一つの軸に沿って隣接するセルをつぎつぎと見たとき（例えば図4.4のCから1軸正方向に並ぶセルC_1, C_2, C_3），そのセルを包含する直方体の集合がCを包含する直方体の集合$\{R_2, R_3\}$とは異なる，Cに最も近いセルすなわちC_2を，Cに**弱隣接する**（weak adjacent）セルと呼ぶ．各軸の正および負方向について同様に定義すると，図4.4の場合，Cに弱隣接するセルは全部でC_2, C', C'', C'''となり，$2m = 4$個ある．この弱隣接の考え方がSAT問題にくらべ拡張された点である．

多次元直方体被覆問題に拡張されたLS法を図4.5に示す．ここで，$n(C)$はセルCを包含する直方体の数である．

一般に，LS法では，探索中の解の候補Cが局所最適解に陥った場合（つまり$n(C') \geq n(C)$の場合）に次の候補をどのように定めるかが効率化のために重要である．第4.1節で紹介し

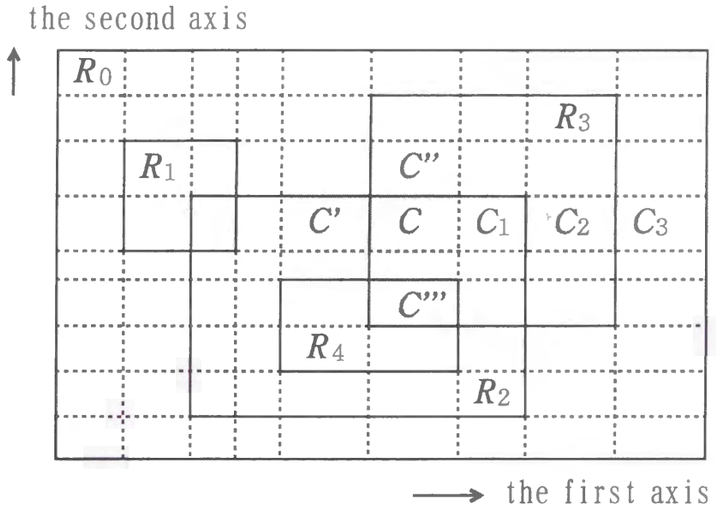


図4.4: 弱隣接セル

step1: ランダムにセルCを一つ定め，$n(C)$を計算する;
step2: $n(C) = 0$のとき，R_0は被覆されていないとして終了;
step3: Cと弱隣接するセル$C_1, \ldots, C_{2m}$を求め，それらの中で$n(C_i)$が最小なセルを求める（これをC'と記す）；$n(C') < n(C)$ならばC'を新しいCとしてstep2へ；$n(C') \geq n(C)$ならばstep1へ;

図4.5: 多次元直方体被覆問題に拡張されたLS法

た文献[SLM92]のSAT問題のためのLS法は，探索中のセルが局所最適解に陥った場合でも$n(C')$を与えるセルC'の一つに遷移して探索を継続し，このような探索を予め定められた回数だけ繰り返しても解が見つからない場合のみ，次の候補となるセルをランダムに作りなおす（step1）．一方，上記のLS法は，局所最適解に陥った場合には次の候補をすぐにランダムに作りなおす（step1）という最も基本的な方法を採用している．

局所最適解に陥った場合の対応を変更する以外に，$n(C_i)$の計算を行う対象となるセルC_iを，Cから各軸の正方向および負方向に並ぶすべてのセル（例えば，図4.4の1軸正方向ではC_1, C_2, C_3）に変更したLS法を考えることもできる．しかし，実験の結果，このLS法については弱隣接セルのみを調べる上記のLS法より非効率であった．第4.5節の実験では，局所最適解に陥った場合の対応が最も基本的であり，かつ，弱隣接セルのみを調べる，上記の局所探索法を使用する．

4.4 新しい解法

4.4.1 基本操作

領域DMとDMに包含された複数の直方体がある．DM全体が被覆されているか否かの状態を変えないようにしながら直方体を変形するための三つの操作を以下に与える．

(1) **切り取り変形**（cut transformation）

図4.6(a)のように，直方体R''のある境界平面により直方体R'を二つの直方体R^-とR^+に分割したときに，R''がその一方（例えばR^+）を包含するようなR''が存在する場合，R'をR^-に縮小することができる．特に，R''がR'全体を包含する場合は，R'全体を消去することができる．この操作を**切り取り変形**（cut transformation）と呼ぶ．

(2) **面移動変形**（move transformation）

図4.6(c)のように，直方体$R' = R(b'_1, \ldots, b'_m, e'_1, \ldots, e'_m)$の$2m$個の周囲面のうち，領域$DM$の周囲面に接していないある面$PL$（平面$x_h = e'_h$と$R'$の共通部分とする）に注目し，その面を$h$軸負方向に移動することにより，直方体$R'$を縮小することを考える（$R'$の周囲面$x_h = b'_h$を$h$軸正方向に移動することを考えることもできる）．周囲面PLとの共通部分が面積をもつ直方体を例えば$R''_1 = R(b''_{11}, \ldots, b''_{1m}, e''_{11}, \ldots, e''_{1m})$, $R''_2 = R(b''_{21}, \ldots, b''_{2m}, e''_{21}, \ldots, e''_{2m})$, $R''_3 = R(b''_{31}, \ldots, b''_{3m}, e''_{31}, \ldots, e''_{3m})$とする．また，直方体$R'$および直方体$R''_1$, R''_2, R''_3の開始h座標$b'_h, b''_{1h}, b''_{2h}, b''_{3h}$の中で最も大きいものを$b''_{*h}(= \max\{b'_h, b''_{1h}, b''_{2h}, b''_{3h}\})$と

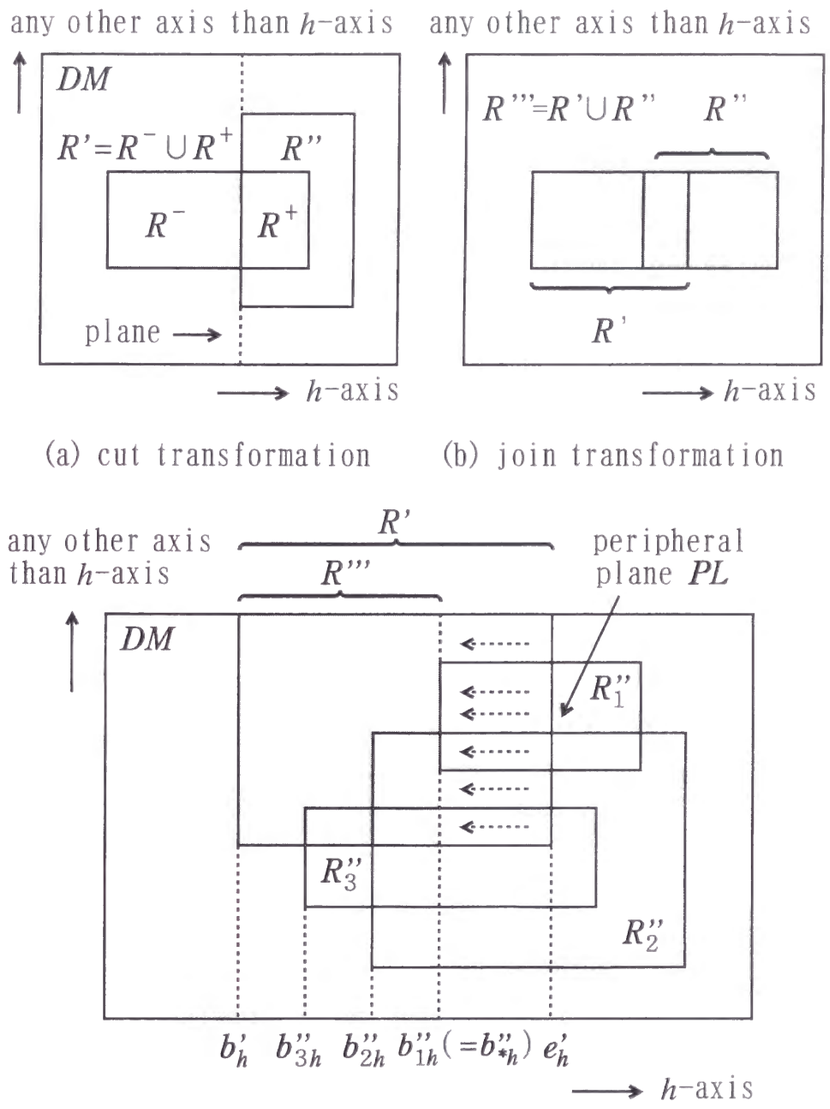


図4.6: 直方体の変形方法

step1: $PST := \{(R_0, \{R_1, \ldots, R_n\})\}$;
step2: $PST = \emptyset$のとき，R_0は被覆されているとして終了;
step3: PSTから部分問題を一つ（(DM, RS)と記す）取り出し，
　step3.1: 切り取り変形可能な直方体$R'(\in RS)$を探す; 見つかればR'を切り取り変形してRSを更新しstep3.1へ;
　step3.2: 面移動変形可能な直方体$R'(\in RS)$を探す; 見つかればR'を面移動変形してRSを更新しstep3.1へ;
　step3.3: 結合変形可能な直方体$R'(\in RS)$と$R''(\in RS)$を探す; 見つかればR'とR''を結合変形してRSを更新しstep3.1へ; 見つからなければstep4へ;
step4: **(ケース1)** $DM = R'$である直方体$R'(\in RS)$が存在する場合，step2へ;
　(ケース2) $RS \neq \emptyset$の場合，単一節分割または純リテラル分割 または単純分割によりDMを小領域DM', DM'''に分割し，$RS = \{R'_1, \ldots, R'_h\}$の各直方体と$DM'(DM''')$との共通部分$R''_i = R'_i \cap DM'(R'''_i = R'_i \cap DM''')$を求め，部分問題$(DM', \{R''_1, \ldots, R''_h\})$と$(DM''', \{R'''_1, \ldots, R'''_h\})$を$PST$に加え，step2へ;
　(ケース3) $RS = \emptyset$の場合，R_0は被覆されていないとして終了;

図4.7: MDP(∞, ∞)法

記す．このとき，周囲面PLを$x_h = b''_{*h}$まで移動することによりR'を縮小しても，DM全体が被覆されているか否かの状態は変化しない．特に，$b''_{*h} = b'_h$の場合，あるいは，周囲面PLとの共通部分が面積をもつ直方体が存在しない場合は，R'全体を消去することができる．この操作を**面移動変形**（move transformation）と呼ぶ．

(3) **結合変形**（join transformation）

図4.6(b)のように，直方体R'と直方体R''がある座標h以外は等しく，かつ，R'とR''の共通部分が体積または面積をもつとき，二つの直方体R'とR''を一つの直方体$R''' = R' \cup R''$に置き換えることができる．この操作を**結合変形**（join transformation）と呼ぶ．

4.4.2 MDP法

MDP法の一つであるMDP(∞, ∞)法を図4.7に与える．ここで，PSTは部分問題を入れるスタックである．

図4.8は$m = 2, n = 4$の問題例にMDP(∞, ∞)法を適用した例である．この例では，(a)，(b)，...，(g)の順に処理が進み，(g)においてDM'が被覆されていないことが分かり，MDP(∞, ∞)法はR_0が被覆されていないとして停止する．

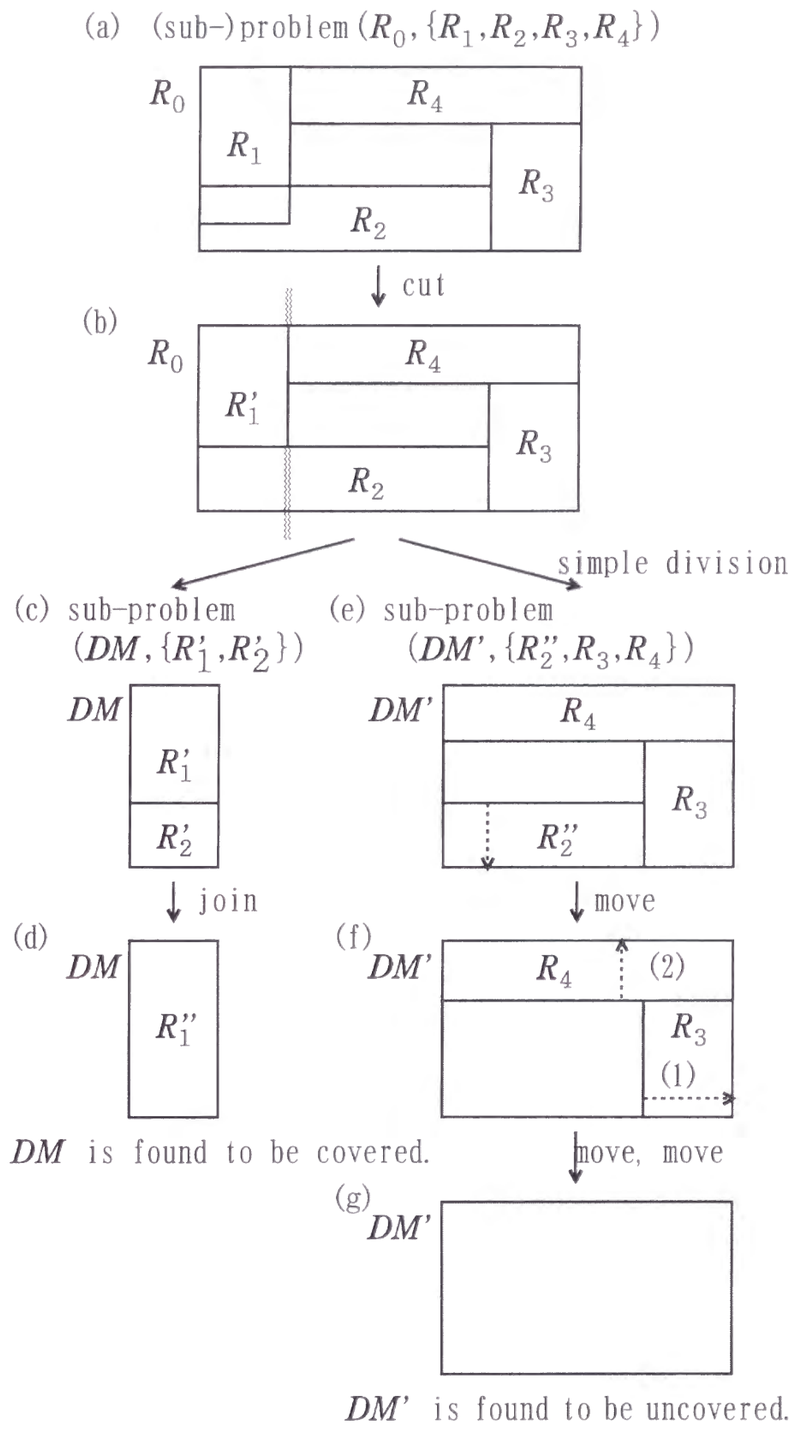


図4.8: MDP法の処理例

各変形操作，分割の役割について述べる．図4.8(e)の部分問題$(DM', \{R_2'', R_3, R_4\})$のように，被覆されていない大きな部分がある場合，面移動変形を繰り返し適用することにより部分問題に含まれるすべての直方体を消去できる（すなわち，その部分問題が被覆されていないと判定できる）可能性がある．また，図4.8(c)の部分問題$(DM, \{R_1', R_2'\})$のように，部分問題が被覆されている場合，結合変形を繰り返し適用することにより部分問題の領域DMに等しい大きな直方体を作れる（すなわち，その部分問題が被覆されていると判定できる）可能性がある．但し，図4.8(b)のように，面移動変形も結合変形もそのままでは適用できない場合があり，その場合には問題を部分問題に分割するなどの操作が必要となる．

一般に，問題の分割以外に，(i) 切り取り，結合変形は面移動変形の適用を容易にし，(ii) 切り取り，面移動変形は結合変形の適用を容易にする．

(i)について説明する．直方体R'の周囲面PL'と直方体R''の周囲面PL''が接しているためにR'の面移動変形が適用できない場合を考える．PL'とPL''の共通部分をPL'''と記す．切り取り変形により，ある直方体RがPL'''を含むR''の一部またはPL'''を含むR'の一部を切り取るならば，または，結合変形により，別のある直方体RとR''が結合して面PL''が消去されるならば，PL'を移動するR'の面移動変形が可能になる．

(ii)について説明する．直方体R'の開始h座標b_h'と直方体R''の開始h座標b_h''が異なるためにR'とR''が結合できない場合を考える．R'の周囲面$x_h = b_h'$をPL'，R''の周囲面$x_h = b_h''$をPL''，更に，ある別の直方体Rの終了h座標をe_h，Rの周囲面$x_h = e_h$をPLと記す．切り取り変形により，RがPL'を含むR'の一部およびPL''を含むR''の一部を切り取るならば，または，面移動変形により，R'の周囲面PL'およびR''の周囲面PL''がRの周囲面PLと接するまで移動するならば，R'の開始h座標およびR''の開始h座標はe_hとなり一致するので，R'とR''の結合変形が可能になる．

MDP(∞, ∞)法の停止性について述べる．切り取り，面移動変形では各直方体の体積は減少する．また，結合変形ではできた直方体$R'''(= R' \cup R'')$の体積はR'の体積とR''の体積の和より増加することはなく，直方体の数は減少する．従って，各部分問題において適用される変形回数は有限である．元問題$(R_0, \{R_1, \ldots, R_n\})$を根，生成される部分問題を節点とする木$T$を考える．各部分問題において各直方体の境界平面は元問題の直方体$R_1, \ldots, R_n$の境界平面のいづれかと一致している．従って，全領域R_0を$R_1, \ldots, R_n$の境界平面により$(2n+1)^m$個のセルにまで細分して部分問題を作れば，その部分問題の被覆状態は必ず判定

できる．このとき，Tの節点数は$2(2n+1)^m - 1$以下である．各部分問題における変形回数の有限性，および，部分問題数の有限性によりMDP(∞, ∞)法は必ず停止する．

MDP(∞, ∞)法の正当性について述べる．各部分問題(DM, RS)において，切り取り，面移動，結合の一連の変形の前の(DM, RS)と一連の変形の後の(DM, RS)を比べた場合，変形前の(DM, RS)が被覆されている（いない）ならば，各直方体の変形にも関わらず，変形後の(DM, RS)も被覆されている（いない）．また，一連の変形後の(DM, RS)を部分問題(DM', RS')と(DM''', RS''')に分割する場合，(DM, RS)が被覆されているならば，(DM', RS')と(DM''', RS''')は両方とも被覆されており，(DM, RS)が被覆されていないならば，(DM', RS')と(DM''', RS''')の少なくとも一方は被覆されていない．(但し，一連の変形前の(DM, RS)において領域DM'''が被覆されていたにも関わらず，(DM''', RS''')が被覆されなくなることはある．）この議論を上に述べた木Tの根（すなわち，元問題$(R_0, \{R_1, \ldots, R_n\})$）から初めてすべての節点（すなわち，部分問題）について繰り返すと，元問題が被覆されているならばTのすべての葉節点は被覆されており，元問題が被覆されていないならば葉節点の少なくとも一つは被覆されていないことになる．故に，MDP(∞, ∞)法は問題を正しく解く．

MDP(∞, ∞)法の効率化のための工夫について述べる．MDP(∞, ∞)法のstep3.1（3.2, 3.3）では，切り取り（面移動，結合）変形可能な直方体を探すために，消去されずに残っている各直方体に対し，順番に，切り取り（面移動，結合）変形可能か否かを検査（これらを切り取り（面移動，結合）**変形テスト**と呼ぶ）していく．これらの変形テストにかかる計算量がMDP(∞, ∞)法の主な計算量である．そのため，効率化のために，無駄なテストをなるべく行わないように工夫している．具体的には，ある直方体が過去の変形テストで変形不可能と判定された場合，その後，変形できる可能性が生じる（但し，実際には変形できないかもしれないが）まで，その直方体の変形テストを行わないようにしている．例えば，直方体Rの面移動変形が不可能と判定された場合，その後，R自身が切り取り，結合，分割（step4のケース2）により変形されるか，または，Rと接していた少なくとも一つの直方体が切り取り，面移動，結合により変形されRと接しなくなるまで，Rの面移動変形テストは行わない．

MDP(∞, ∞)法の変形操作の回数を制限して得られるMDP法について述べる．MDP(∞, ∞)法では，各部分問題において，すべての直方体が変形できなくなるまで変形を進めている．しかし，各部分問題における変形テストの回数に制限を設け，変形可能な直方体が

あっても変形を行わないようにすることもできる．設定する制限の強さにより，例えば以下に述べるようなMDP法を定義することができる．一つの部分問題における，各直方体Rに対する切り取り（面移動，結合）変形テストの回数およびRを変形して得られる各直方体に対する切り取り（面移動，結合）変形テストの回数の合計を$ct(R)$ $(mt(R), jt(R))$と記し，また，一つの部分問題におけるすべての切り取り（面移動，結合）変形テストの回数を$ct(*)$ $(mt(*), jt(*))$と記す．MDP(a,b)法は，各部分問題における$ct(R), mt(R), jt(R)$の各上限をaに制限し，かつ，$ct(*), mt(*), jt(*)$の各上限もbに制限したMDP(∞,∞)法である．なお，MDP$(0,0)$法はDP法に一致する．

MDP法の1部分問題当たりの最悪時間量を示す．1回の切り取り（面移動，結合）変形テストの時間量は，一つの直方体と他のすべての直方体との関係を調べるため，$O(nm)$である．従って，MDP(a,b)法の1部分問題当たりの最悪時間量は，$b < an$の場合，$O(bnm)$となり，$b \geq an$の場合，$O(an^2m)$となる．また，結合によりできた直方体を別の直方体と考えると，一つの直方体Rに対する$ct(R), mt(R), jt(R)$は先の工夫により，各々，$O(n)$となるので，MDP(∞,∞)法の1部分問題当たりの最悪時間量は$O(n^3m)$となる．なお，DP法の1部分問題当たりの最悪時間量，および，LS法で一つのセルが被覆されているか否かを検査する最悪時間量は$O(nm)$である．

4.5 実験

第4.1節に述べたように，3種類の方法で問題例を作り実験を行う．分割型問題例，拡大型問題例，ランダム型問題例の実験の順に説明し，最後に，これら三つの実験の結果をまとめる．

4.5.1 分割型問題例の実験

分割型問題例 $I = (R_0, \{R_1, \ldots, R_n\})$は次のように生成する．図4.9のように，(1) 同じ全領域R_0をs個（$R_0^1, \ldots, R_0^s$と記す）用意する．(2) 次に，各領域$R_0^j (= R_0), j = 1, \ldots, s$, ごとに，$R_0^j$をランダムに分割して$f$個の直方体を作る．ここで，$R_0^j$の分割は次のように行う．分割の途中段階で得られた直方体の集合を$A = \{R'_1, \ldots, R'_u\}$（ここで，$R_0^j = R'_1 \cup \cdots \cup R'_u$, $i \neq j$のとき$R'_i \cap R'_j = \emptyset$）として，更に分割を進める場合には，集合$A$の中から直方体をランダムに一つ取り出し（$R'_u$とする），ある軸に垂直な，$R'_u$と交わる$t$個の平面をランダムに

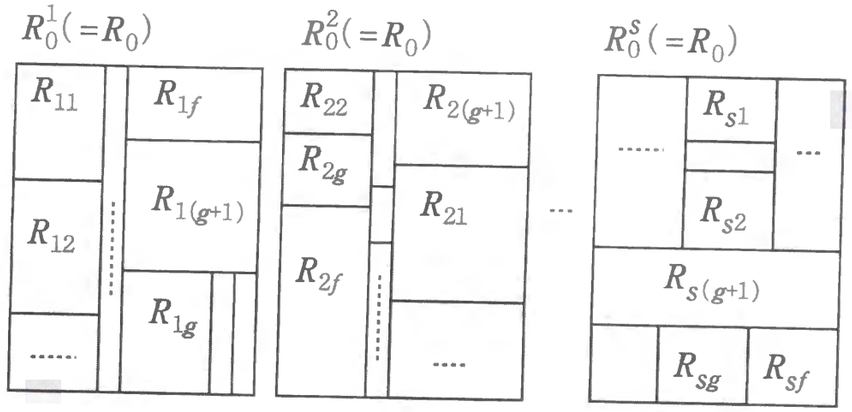


図4.9: 分割型問題例の作り方

選び，これらの平面によりR'_uを分割して直方体$R''_1, \ldots, R''_{t+1}$を作り，これらをAに加える（$A = \{R'_1, \ldots, R'_{u-1}, R''_1, \ldots, R''_{t+1}\}$）．この操作を$A = \{R_0^j\}$から初めて$A$の中の直方体の数が$f$になるまで続ける．(3) 最後に，$g = n/s$として，各$R_0^j$ごとに，(2) で作られた$f$個の直方体の中からランダムに$g$個の直方体（$R_{j1}, \ldots, R_{jg}$と記す）を選び，すべての$R_0^j$から選ばれた直方体を集め，問題例$I$の直方体$R_1, \ldots, R_n$とする（$\{R_1, \ldots, R_n\} = \cup_{j=1}^{s} \{R_{j1}, \ldots, R_{jg}\}$）．なお，$n/s(= g)$が整数でない場合には，$g$を$\lceil n/s \rceil$または$\lfloor n/s \rfloor$として全体で$n$個の直方体を選ぶようにする．また，例えば$s = 2$の問題例と$s = 3$の問題例の中間の性質をもった問題例，例えば$s = 2.2$の性質をもった問題例を作るために，$s$が整数でない場合には次のようにする．全領域$R_0$を$\lceil s \rceil$個用意し，その内の一つ（$R_0^{\lceil s \rceil}$とする）からは$g$の代わりに$g' = \lfloor n(s - \lfloor s \rfloor)/s \rfloor$個の直方体を選び，残りの$\lfloor s \rfloor$個の全領域$R_0^1, \ldots, R_0^{\lfloor s \rfloor}$の各々からは，$g$の代わりに$g'' = (n - g')/\lfloor s \rfloor$個（$(n - g')/\lfloor s \rfloor$が整数でない場合は$g'' = \lceil (n - g')/\lfloor s \rfloor \rceil$または$\lfloor (n - g')/\lfloor s \rfloor \rfloor$個）の直方体を選ぶようにする．

$1 \leq n/f \leq s \leq n$とする．問題例Iにおいて，直方体$R_1, \ldots, R_n$が被覆する部分$B = \cup_{i=1,n} R_i$の平均体積は$\overline{|B|} = |R_0|(1 - (1 - n/(sf))^s)$となる（但し，$n/s$または$s$が整数でない場合は近似値である）．

式$w = n/f$により定義されるwは$(\sum_{i=1}^{n} |R_i|)/|R_0|$の平均値を表しており，**重複度**（duplication degree）と呼ぶ．wは全領域R_0内の各点を包含する直方体の平均数であり，wが大きいほど直方体$R_1, \ldots, R_n$の重複の度合が大きい．また，式$r = s/w$により定義され

る$r(1 \leq r \leq n/w)$を**ランダム度**（randomness degree）と呼ぶ．rは直方体$R_1, \ldots, R_n$の配置の乱雑度を表しており，rが大きいほど，同じR_0^jから選ばれる直方体の数gまたはg''が小さくなるので，直方体の配置はランダムになる．また，rが大きくなるにつれて，平均体積$\overline{|B|} = |R_0|(1-(1-1/r)^{wr})$は減少し，被覆されていない部分が大きくなる．すなわち，$r=1$のとき，gまたはg''は(gまたはg'')$= n/w$で最大であり（このとき，問題例Iは被覆され，$\overline{|B|} = |R_0|$であり），rが増加するにつれて，gまたはg''は減少し（このとき，$\overline{|B|}$は減少し），$r = n/w$のとき，gは$g=1$で（このときは$s=n$が整数であるのでgのみ）最小となり，直方体$R_1, \ldots, R_n$の配置は完全にランダムになる（このとき，$\overline{|B|}$は最小となる）．このように，分割型問題例では，ランダム度rを変化させることにより，重複度wを一定に保ったまま，被覆された問題例から被覆されていない問題例まで作ることができる．

実験では，全領域R_0の1辺の長さ$L=2000$，次元$m=30$，直方体の数$n=300$，分割のときに使う平面の数$t=2$，重複度$w = 30, 27.2, 25, 23, 20, 18.7, 16.6, 14.2, 12, 10, 8.1, 6, 4, 2, 1$（すなわち，分割数$f = n/w = 10, 11, 12, 13, 15, 16, 18, 21, 25, 30, 37, 50, 75, 150, 300$），ランダム度$r = 1.00, 1.01, 1.05, 1.1, 1.2, 1.3, \ldots, 2.0, 3.0, 4.0, 5.0, 10, 20, 30, 40, 50, 300$，および$n/w$（但し，条件$r \leq n/w$を満たす$r$のみ）の計289組の各組合せに対し3例づつ問題例を作り，合計869例の問題例を作った．これらの各問題例に対し，20 mipsのリスクワークステーション上で計算時間の上限を1000秒として，DP法，LS法，MDP(∞,∞)法，MDP(2,30)法の各方法を実行した．

図4.10に$w=2$の場合の，図4.11に$w=10$の場合の各方法の平均計算時間を示す．但し，LS法については，被覆された問題例では停止しないので，被覆されていなかった問題例に対する実行結果のみを示す．グラフの縦軸は3例の平均計算時間を示し，グラフの上部の“タイムアウト”は3例中少なくとも1例がタイムアウトしたことを示す（2例が短時間で解けたにも関わらず，1例がタイムアウトした場合もあった）．目盛りは対数目盛になっている．また，グラフの横軸はランダム度を示すが，目盛りは線形目盛りでも対数目盛りでもなく，値の小さな部分がより細かい特殊な目盛りになっている．横軸の“covered”（“uncovered”）は3例すべてが被覆されていた（いなかった）ことを，“covered”と“uncovered”の間は3例中に被覆された問題例と被覆されていない問題例が混じっていたことを示す．図4.10，図4.11のいずれにおいても，DP法やLS法は，rの大きな（すなわち，被覆されていない部分が大きな）問題例に対しては効率的であるが，rの小さな（すなわち，被覆されていない部分の小さな　被覆された）問題例に対してはタイムアウトしている．一方，MDP(∞,∞)

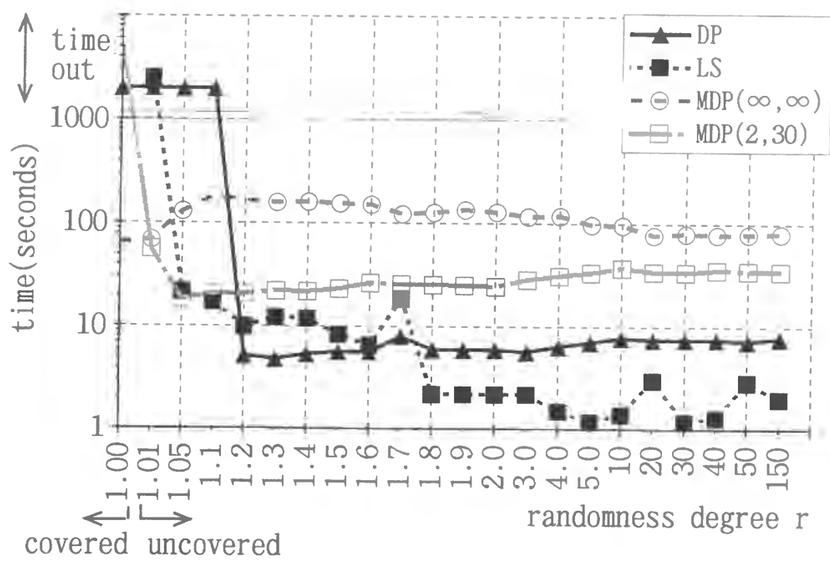


図4.10: 分割型問題例に対する平均計算時間（$w=2$の場合）

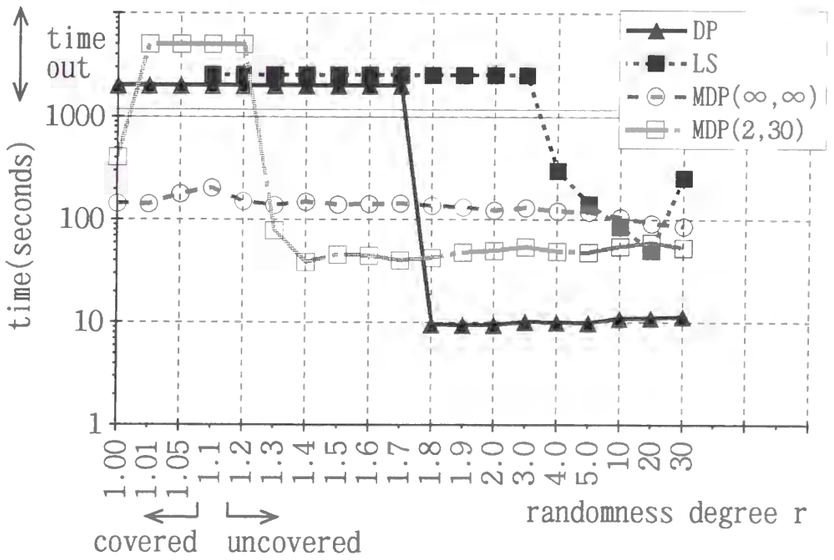


図4.11: 分割型問題例に対する平均計算時間（$w=10$の場合）

法は，r全域に渡ってタイムアウトすることなく問題例を解いている．また，MDP(2,30)法はDP法とMDP(∞,∞)法の中間的な挙動をしている．

なお，重複度wを大きくしていった場合，LS法は相変わらずタイムアウトし，DP法は$w \geq 27.2$ではほとんどタイムアウトしなくなり，MDP(∞,∞)法は$w \geq 16.6$ではタイムアウトするようになり，MDP(2,30)法は$w = 16.6, 18.7, 20$ではほとんどタイムアウトしなくなったが，$w \geq 23$では再びタイムアウトするようになった．各組合せ(r,w)に対し平均計算時間が最も小さい方法を図4.12に示す．図の点が記されていない組合せ(r,w)では，3例すべてを解いた解法がなかった．また，各重複度wに対応する行の右端の点は最大ランダム度$r = n/w$における実験結果であるが，その点のランダム度（すなわち最大ランダム度）だけは，横軸の値と異なる場合があるため，点の右に数値で示した．例えば，$w = 2$の行の右端の点の右の数値150$(= 300/2)$はその点のランダム度である．

図4.10，図4.11，図4.12に示したように，計算時間においては，全領域の全体またはほとんどが被覆されていて，かつ，重複度が低い（$w \leq 23$）問題例に対して，MDP法（MDP(∞,∞)法またはMDP(2,30)法）はDP法およびLS法より効率的であった．

図4.13に$w = 10$の場合の各方法の平均部分問題数を示す．但し，各ランダム度に対し3例すべてが解けた場合のみ平均値を示した．また，LS法については，被覆されていない問題例のすべてが解けた場合に，被覆されているか否かを検査したセルCの数の平均値を示した．すべての重複度における実験の結果，部分問題数においては，重複度が大変低い（$w \leq 10$）問題例に対してはMDP(∞,∞)法が，また，それ以外の重複度が低い（$12 \leq w \leq 23$）問題例に対してはMDP(∞,∞)法またはMDP(2,30)法が，ランダム度rのほぼ全域に渡って，DP法およびLS法より効率的であった．

4.5.2 拡大型問題例の実験

分割型問題例では，ランダム度rが1に近いと，一つの全領域R_0をf個に分割してできた直方体のほとんどが問題例の直方体として選ばれ，問題例の直方体が整然と並ぶ．第4.5.1節の分割型問題例の実験結果はこの特殊な性質に起因している可能性もある．そこで，分割型問題例のこの性質を軽減した問題例（拡大型問題例と呼ぶ）を作って実験を行い，拡大型問題例に対しても分割型問題例の場合と同様の結果が得られるか調べる．

拡大型問題例は，次のように，分割型問題例の直方体を更に少しランダムに拡大して作る．分割型問題例$I = (R_0, \{R_1, \ldots, R_n\})$の各直方体$R_i, i = 1, \ldots, n,$に対し，全領域$R_0$の

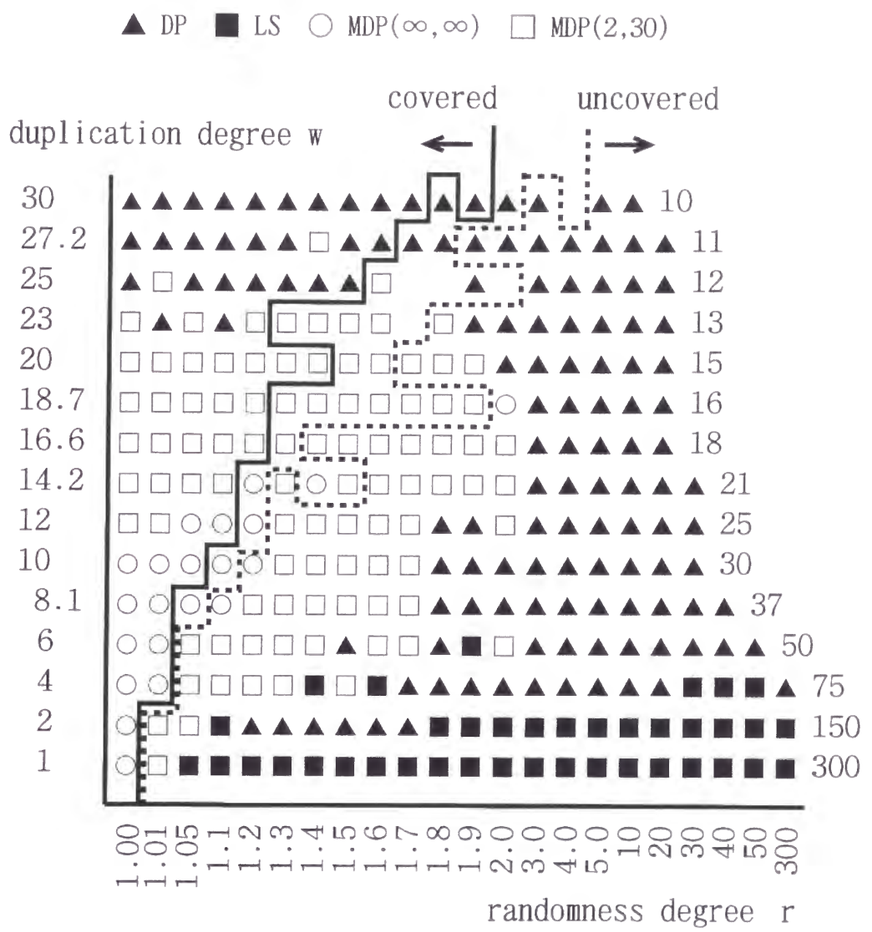


図4.12: 分割型問題例の実験において平均計算時間が最も小さい方法

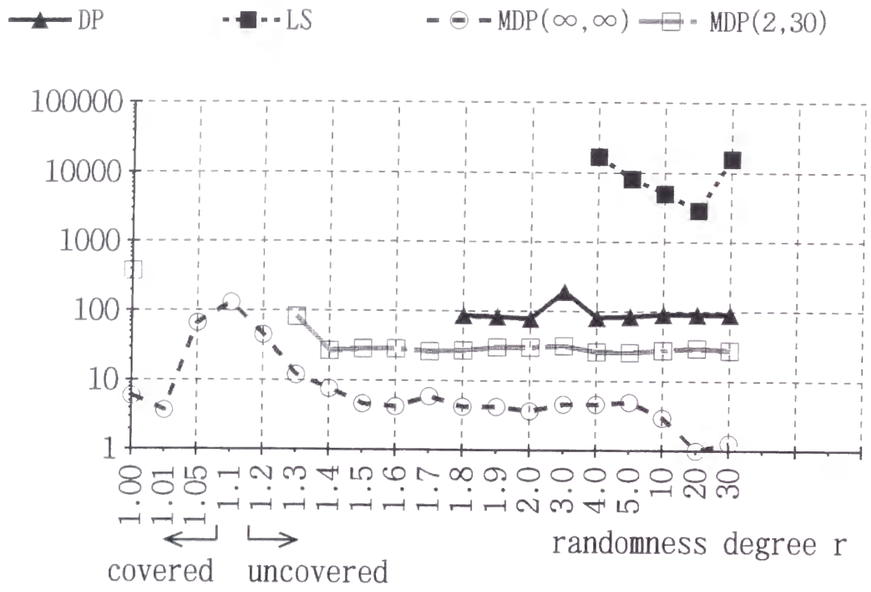


図 4.13: 分割型問題例に対する平均部分問題数（$w = 10$ の場合）

周囲面に接していない（R_i の）各周囲面 PL を R_i の体積が増加する方向に 0 から max_h の範囲でランダムに移動して直方体 R'_i を作り，拡大型問題例 $I' = (R_0, \{R'_1, \ldots, R'_n\})$ とする．ここで，R_0 の周囲面に接している，R_i の周囲面 PL は移動しない．

先の実験により作られた $w = 2, 10, 30$ の分割型問題例に対し，$max\_h = 100$ として拡大型問題例を作り，MDP(∞, ∞) 法および DP 法を適用した（MDP(2,30) 法と LS 法は省略した）．$w = 10$ の場合の分割型問題例より作った拡大型問題例に対するこれら二つの解法の平均計算時間を図 4.14に，平均部分問題数を図 4.15に示す．図 4.14と図 4.11，図 4.15と図 4.13を比べると，計算時間および部分問題数のどちらにおいても，二つの解法の効率は拡大型問題例と分割型問題例でそれほど変わっていない．$w = 2, 10, 30$ の場合のいずれの実験においても，このように，MDP(∞, ∞) 法と DP 法の優劣は分割型問題例の場合とほぼ同じであり，重複度が低く（$w = 2, 10$），かつ，全領域の全体または全領域のほとんどの部分が被覆された問題例に対しては，MDP(∞, ∞) 法は DP 法より効率的であった．

なお，w, r, m を $w = 10, r = 1.2, m = 30$ だけにし，n を $n = 100, 200, 300, 600, 900, 1200$ に変化させて拡大型問題例を作り，また，w, r, n を $w = 10, r = 1.2, n = 300$ だけにし，m を $m = 10, 20, 30, 60, 90, 120$ に変化させて拡大型問題例を作り，これらの問題例に対して

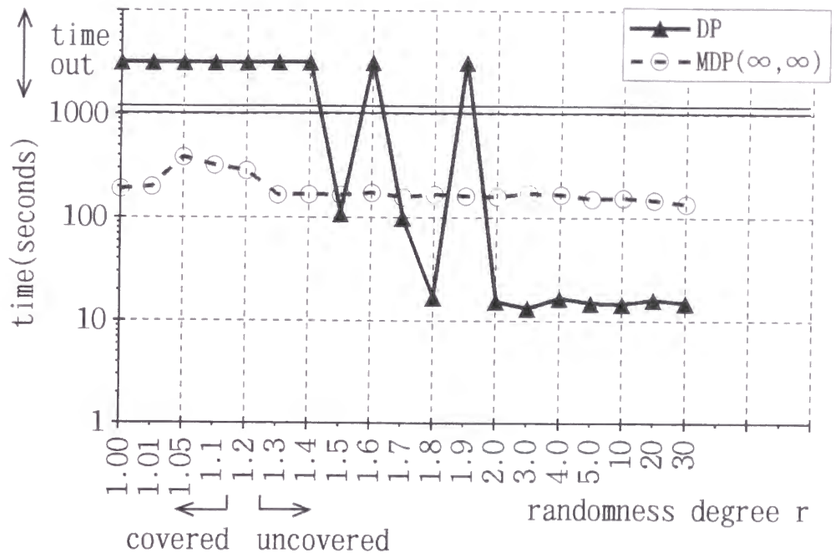


図 4.14: 拡大型問題例に対する平均計算時間

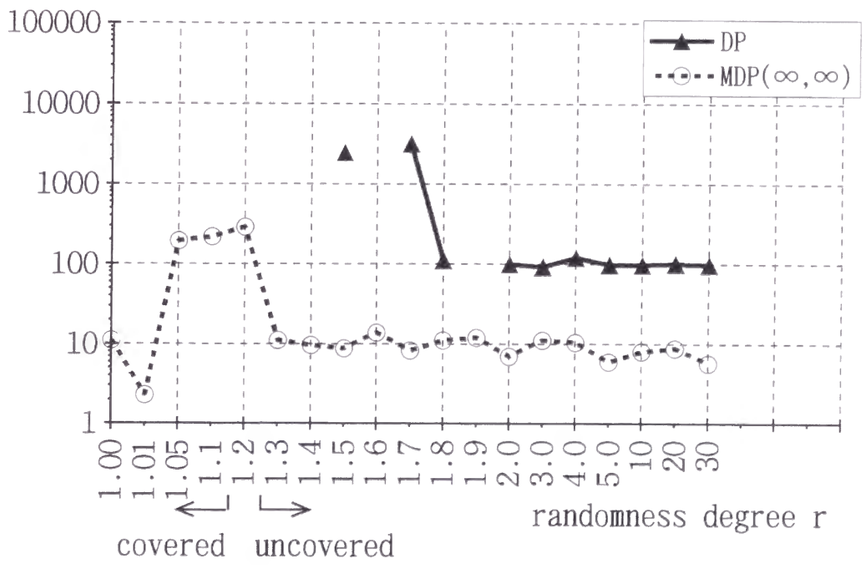


図 4.15: 拡大型問題例に対する平均部分問題数

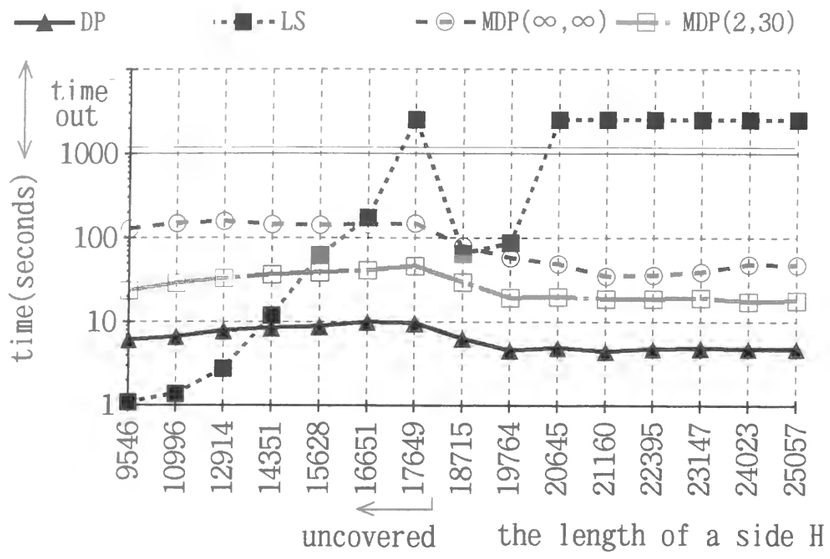


図 4.16: ランダム型問題例に対する平均計算時間

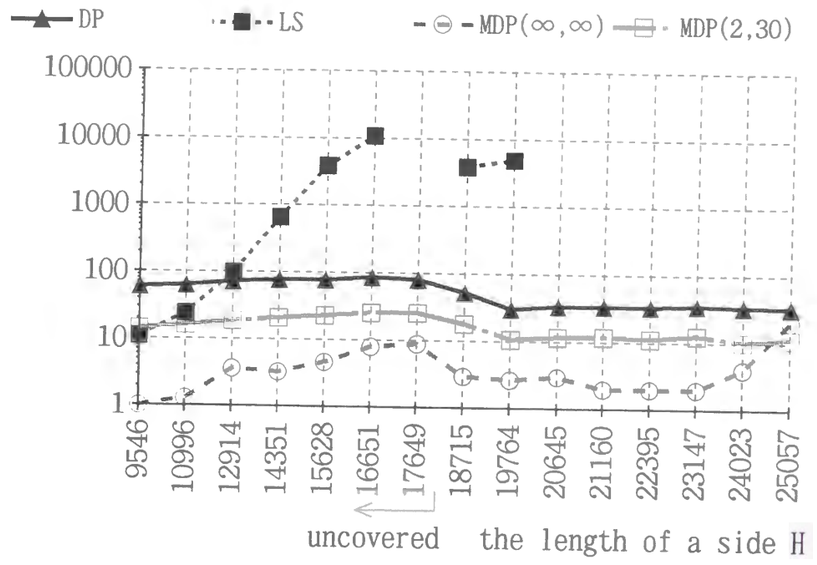


図 4.17: ランダム型問題例に対する平均部分問題数

も，タイムアウト時間を大きくとり，MDP(∞,∞)法およびDP法を適用して実験を行った．この実験では，二つの解法の計算時間はnまたはmの増加につれて指数関数的に増大したが，この実験おいても，二つの解法の優劣は他の実験の場合とほぼ同じであった．

以上，拡大型問題例に対しても分割型問題例の場合と同様の結果が得られた．

4.5.3 ランダム型問題例の実験

充足可能性問題の研究においてよく使われるランダムSAT問題例の作り方を多次元直方体被覆問題用に拡張してランダム型問題例を作り，この問題例に対しても実験を行う．

1辺の長さHのm次元直方体$R'_i, i = 1,\ldots,n,$を領域$\{(x_1,\ldots,x_m) \mid -H+1 \leq x_i \leq L+H-1, i = 1,\ldots,m\}$の内部にランダムに配置し，$R'_i$と全領域$R_0 = \{(x_1,\ldots,x_m) \mid 0 \leq x_i \leq L, i = 1,\ldots,m\}$の共通部分$R_i = R'_i \cap R_0$を集めて，**ランダム型問題例**$I = (R_0, \{R_1,\ldots,R_n\})$とする．

実験では，$m = 30, n = 300$とし，重複度$w = (\sum_{i=1}^{n} |R_i|)/|R_0|$の平均値が分割型問題例の実験における重複度$w = 1,2,\ldots,30$と同じになるように$H$を定め，各$H$に対し3例づつ問題例を作り，DP法，LS法，MDP(∞,∞)法，MDP(2,30)法を実行した．平均計算時間を図4.16に，平均部分問題数を図4.17に示す．ランダム型問題例に対する各解法の実行結果は同じ重複度の最大ランダム度をもつ分割型問題例に対する実行結果とよく似ている．MDP(∞,∞)法，MDP(2,30)法は，部分問題数においてはDP法，LS法より効率的であったが（図4.17），計算時間において非効率であった（図4.16）．

4.5.4 実験のまとめ

以上の3種類の問題例に対する実験より，

1. 提案したMDP法（MDP(∞,∞)法およびMDP(2,30)法）は，全領域の全体または全領域のほとんどの部分が被覆されていて，かつ，直方体の重複の度合が低い問題例に対し，従来のDP法およびLS法より効率的である，

2. MDP法，DP法，LS法のいずれも，それ一つだけでは，被覆されていない部分の大きさおよび直方体の重複の程度が様々に異なる問題例のすべてを効率的に解くことはできないが，しかし，例えば図4.12から分かるように，これら三つの解法を併用すれば，そのような問題例の多くを効率的に解くことができる

ことが分かる．1においてMDP法がそのような問題例に対して効率的であるのは，直方体が他の直方体と重なる部分が少ないために面移動変形が急速に進み，また，その結果，結合変形も急速に進むためと思われる．

第4.2節に説明した直積法と多次元直方体被覆問題の関係，および，上の2より，MDP法，DP法，LS法の三つの解法を直積法に組み込むことにより，連続直積問題の多くに対して直積法をより効率化できることが分かる．

4.6 むすび

直積問題を制限した連続直積問題に直積法を適用したときに直積法の内部処理に現れる組合せ問題で，この組合せ問題を効率的に解ければ，その解法を直積法に組み込むことにより，連続直積問題に対する直積法の効率を改善できる組合せ問題（すなわち，多次元直方体被覆問題）について考えた．具体的には，一つのm次元直方体（すなわち，全領域）が複数のm次元直方体の和により被覆されているか否かを判定する問題である．

多次元直方体被覆問題を効率的に解くための解法として，充足可能性問題のためのDavis-Putnum法を直方体群の幾何学的性質を利用して拡張した新しい解法（すなわち，変形Davis-Putnum法）を提案した．新しい解法，および，充足可能性問題のためのDavis-Putnum法と局所探索法を多次元直方体被覆問題に素直に拡張した解法の効率を調べるために計算機実験を行い，その結果，(1) 提案した変形Davis-Putnum法が，全領域の全体または全領域のほとんどの部分が被覆されていて，かつ，直方体の重複の度合が低い問題例に対し，従来のDavis-Putnum法および局所探索法より効率的であること，および，(2) これら三つの解法を併用すれば，様々な性質をもった問題例の多くを効率的に解けること，を示した．

直積法と多次元直方体被覆問題の関係，および，(2) の結果として，変形Davis-Putnum法，Davis-Putnum法，局所探索法の三つの解法を直積法に組み込むことにより，連続直積問題の多くに対して直積法をより効率化できることが示せた．

第5章

一般の問題を効率的に解くための問い合わせ処理法

演繹データベースにおける一般の問題を効率的に解くことを目的に，マジック集合法やその他多くの方法が提案されている．これらの方法は，acyclic sipという考え方に基づいて問い合わせアトム中の拘束を各ルールに伝搬させることにより制約を求め，その制約を使って解の生成に関連しない不要な基礎アトムの生成をできるだけ防ぐことにより，問題を効率的に解く．

本章では，緩和法と呼ぶ新しい問い合わせ処理法を提案する．解の生成に関連する基礎アトムの集合を関連集合と呼ぶ．緩和法は，従来の方法と同様に，制約を使って問題を効率的に解くが，しかし，従来の方法とは異なり，関連集合の緩和という考え方に基づいて制約を求める．マジック集合法の一つであるマジックテンプレート法は，一般の問題を効率的に解くための従来の方法の中で最も適用範囲が広く，datalog問題クラスと一部のホーン問題を解くことができ，かつ，最も効率的である（ただし，アレクサンダーテンプレート法とHCT/R法もマジックテンプレート法と同じ能力を持つ）．マジックテンプレート法と緩和法を比較し，マジックテンプレート法が解くことができる任意の問題を緩和法が少なくとも同じ効率で解けること，マジックテンプレート法が解けるある問題を緩和法がより効率的に解けること，マジックテンプレート法が解けないある問題を緩和法が効率的に解けること，を示す．なお，緩和法の適用範囲はdatalog問題クラスと一部のホーン問題であるが，ホーン問題部分がマジックテンプレート法より広い．

5.1 はじめに

演繹データベースにおける問題を効率的に解くことを目的に適用範囲や効率の異なる様々な方法が提案されている．これらの方法は，特定の問題を対象とした方法，一般の問題ほど広くはないが比較的広い範囲の問題を対象とした方法，一般の問題を対象とした方法の三つに大きく分類することができる．特定の問題を対象とした方法には，例えば，右線形問題を対象としたNRSU法[NRSU89b]，CRL法[KRS90]，CORL法[MP91]や，同世代問題を対象とした計数法[BMSU86, BR86]（第1.4.2節を参照），逆計数法[BMSU86, BR86]，HaNa法[HN91]（第2章を参照），拡張HaNa法（第2章を参照）等がある．比較的広い範囲の問題を対象とした方法には，直積問題を対象とした直積法（第3章を参照）等がある．一般の問題を対象とした方法には，マジックテンプレート法[Ram88]，アレクサンダーテンプレート法[Sek89]，HCT/R法[MYHI89]等がある．特定または比較的広い範囲の問題を対象とした方法は一般の問題を対象とした方法より問題を効率的に解くことが多いが，しかし，前者の方法の適用範囲はdatalog問題の一部に限られている．故に，特定または比較的広い範囲の問題を対象とした方法を適用できない問題を解く場合には一般の問題を対象とした方法が必要となる．

本章では，一般の問題を効率的に解くことを目的に，**緩和法**（relaxation method）と呼ぶ新しい方法を提案する．

本節では，初めに，一般の問題を効率的に解くための従来の方法について説明し（すでに第1.4.3節にて説明したが，本章の理解を容易にするために，再度，簡単に説明する），次に，提案する緩和法について概要を説明し，最後に，本章の構成を述べる．

初めに，従来の方法について説明する．問い合わせ処理の効率化にとって，生成するga集合S_IMPの大きさを小さくすることが重要である．一般の問題のための方法はいずれも制約を求め，制約を使って解の生成に無関係なgaの生成をできるだけ防ぐことで，S_IMPを小さくしている．また，問い合わせ処理の効率化にとって，同じ計算の重複を防ぐことも大切である．大きなルールを複数の小さなルールに分割することで，記憶量の増加と引き替えに，計算の重複を防ぐことができる場合があるので，一般の問題のための方法の一部は制約の利用に加えてルールの分割も併用している（ただし，記憶量が爆発する場合があるので，このような場合には使えない）．

一般の問題を効率的に解くための有名な方法に，**基本マジック集合法**（basic magic sets method）[BMSU86]，**一般化マジック集合法**（generalized magic sets method）[BR87]，**一般化補助マジック集合法**（generalized supplementary magic sets method）[BR87]，**マジックテンプレート法**（magic templates method）[Ram88]がある．これらの方法は総称して**マジック集合法**（magic sets method）と呼ばれている．また，その他の有名な方法に，**アレクサンダー法**（Alexander method）[RLK86]，**アレクサンダーテンプレート法**（Alexander templates method）[Sek89]，**HCT/R法**（horn clause transformation by restrictor）[MYHI89]がある．できるだけ広い範囲の問題に対して優れた制約を作れるように，基本マジック集合法を一般化したものが一般化マジック集合法と一般化補助マジック集合法であり，更に，一般化マジック集合法を一般化したものがマジックテンプレート法である．同様に，アレクサンダー法を一般化したものがアレクサンダーテンプレート法である．一般化補助マジック集合法とアレクサンダー法とアレクサンダーテンプレート法は制約の利用に加えてルールの分割も併用している．マジックテンプレート法とアレクサンダーテンプレート法とHCT/R法は，ルール分割を除いて，同じ能力をもっている．これら三つの方法は，一般の問題を効率的に解くための従来の方法の中で最も適用範囲が広く，datalog問題クラスと一部のホーン問題を解くことができる．また，これら三つの方法は，一般の問題を効率的に解くための従来の方法の中で，最も優れた制約を作れるという意味で，最も効率的である．

一般の問題を効率的に解くための主となる手法は制約の利用であり，また，ルールの分割は必要ならばいずれの方法に対しても容易に導入することができる．故に，以後，優れた制約を作ることに議論を集中する．

準備として，記法や用語を幾つか与えた後，マジック集合法の問題の解き方，正当性，効率について説明する．簡単のため，問題はdatalog問題である，すなわち，算術述語も関数記号も含まないと仮定する．また，上に述べたように，マジック集合法は制約のみを利用し，ルール分割は用いないものとする．なお，アレクサンダー法，アレクサンダーテンプレート法，HCT/R法については陽には説明しないが，これらの方法もマジック集合法とほぼ同じである．

定数を$a, b, c, \ldots$，定数ベクトルを$\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \ldots$，変数を$X, Y, Z, \ldots$，変数ベクトルを$\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \ldots$と記す．問い合わせアトムを$q(\boldsymbol{a}, \boldsymbol{X})?$，ルール集合を$P$，ファクト集合を$D$，演繹データベース（ddbと記す）を$S = (P, D)$，問題を$(q(\boldsymbol{a}, \boldsymbol{X})?, S)$と記す．ddb Sの中の全ての述語名の集合を$N(S)(= \{p, q, r, \ldots, A, B, C, \ldots\})$，$S$の中の全ての定数の集合を$C(S)(= \{a, b, c, \ldots\})$，述語の全引数に定数を代入して得られる全ての基礎アトム（gaと記す）の集

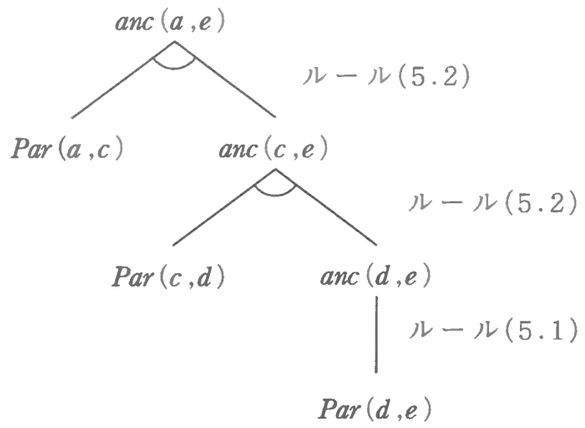


図5.1: 生成木

合，すなわち，**エルブラン基底** (Herbrand base) を $HB(S)(=\{p(a,\ldots,a), p(a,\ldots,b),\ldots, q(a,\ldots,a), q(a,\ldots,b),\ldots\})$，$S$ から生成できるすべての ga の集合，すなわち，意味を $IMP(S)$，問題の解集合を $ANS(q(\boldsymbol{a},\boldsymbol{X})?,S)$ $(=\{q(\boldsymbol{a},\boldsymbol{x}) \mid q(\boldsymbol{a},\boldsymbol{x}) \in IMP(S)\})$ と記す．ddb $S=(P,D)$ のルール集合を

$$P: \quad anc(X,Y) \leftarrow Par(X,Y). \tag{5.1}$$

$$anc(X,Z) \leftarrow Par(X,Y), anc(Y,Z). \tag{5.2}$$

とし，ファクト集合を $D=\{Par(a,c), Par(c,d), Par(d,e),\ldots\}$ とするとき，例えば，図5.1の木のように，ddb より ga を生成することができる．ga が ddb より生成される様子を表すこのような木を**生成木** (generation tree) と呼ぶ．

定義 5.1 ga $s(\boldsymbol{d})(\in IMP(S))$ を生成するある生成木が節点に ga $p(\boldsymbol{c})$ を含むとき ($p(\boldsymbol{c}) \in IMP(S)$ である)，$p(\boldsymbol{c})$ は $s(\boldsymbol{d})$ に**関連している** (relevant) と言い，$(p(\boldsymbol{c}),S) \stackrel{*}{\rightarrow} s(\boldsymbol{d})$ と記す．ある解 $q(\boldsymbol{a},\boldsymbol{b})$ $(\in ANS(q(\boldsymbol{a},\boldsymbol{X})?,S))$ に ga $p(\boldsymbol{e})$ が関連しているとき，$p(\boldsymbol{e})$ は問い合わせアトム $q(\boldsymbol{a},\boldsymbol{X})?$ に**関連している**と言い，$(p(\boldsymbol{e}),S) \stackrel{*}{\rightarrow} q(\boldsymbol{a},\boldsymbol{X})?$ と記す．問い合わせアトム $q(\boldsymbol{a},\boldsymbol{X})?$ に関連している全ての ga の集合を**関連集合** (relevant set) と呼び，$REL(q(\boldsymbol{a},\boldsymbol{X})?,S)$ $(\subseteq IMP(S))$ と記す．

$$REL(q(\boldsymbol{a},\boldsymbol{X})?,S) = \bigcup_{p\in N(S)} \{p(\boldsymbol{e}) \mid (p(\boldsymbol{e}),S) \stackrel{*}{\rightarrow} q(\boldsymbol{a},\boldsymbol{X})?\} \quad \square$$

定義 5.2 緩和条件 (relaxation condition)

$$PREL \supseteq REL(q(\boldsymbol{a},\boldsymbol{X})?,S) \tag{5.3}$$

を満足する任意の ga 集合 $PREL$ $(\subseteq HB(S))$ を**潜在的関連集合** (potentially relevant set) と呼ぶ．$\square$

なお，誤解の恐れのない場合は，$IMP(S), REL(q(\boldsymbol{a},\boldsymbol{X})?,S)$ 等を，() の中を省略して単に，IMP, REL 等と記す．

マジック集合法は，(1) 初め，**acyclic sip** (sideways information passing) [BR87, Ram88] という考え方に基づいて問い合わせアトムおよびルール中の拘束を各ルールに伝搬させることにより，ある小さな ga 集合 MS (**マジック集合**と呼ばれる) を生成し，その MS を使って $PREL$ を定義する．例えば，2引数の IDB 述語 $p(X,Y)$ についての問題があり，$p(X,Y)$ の第1引数についての拘束の集合を表すマジック集合 $MS=\{m\_p(a), m\_p(b),\ldots\}$ を生成したとする．このとき，$PREL$ は

$$PREL \;=\; \{p(x,y) \mid m\_p(x) \in MS, y \text{は任意の定数}\}$$

と定義される．(2) 次に，その $PREL$ を制約として使って，$PREL$ に含まれる ga の生成のみを許しながら，与えられた問題の ddb S から ga を生成していく．このとき，ga 集合 $IMP \cap PREL$ が生成される．(3) 最後に，$IMP \cap PREL$ の中から解である ga を選択し，それらの集合を解集合 ANS とする．ここで，マジック集合法が生成するすべての ga の集合 S_IMP は $S\_IMP = MS \cup (IMP \cap PREL)$ である．

マジック集合法の正当性について説明する．IMP および $PREL$ の性質より $(PREL \cap IMP) \supseteq REL \supseteq ANS$ が成立する．そのため，$IMP \cap PREL$ の中から解を選択しても，ANS を正しく求めることができる．

マジック集合法の効率について説明する．あまり手間をかけずに求めた (すなわち，小さい MS によって定義される) $PREL$ で，REL をうまく近似する (すなわち，$IMP \cap PREL \doteq REL$ である) $PREL$ は優れている，と言う．問題の関連集合 REL の大きさは意味 IMP の大きさに比べ大変小さい ($|REL| << |IMP|$) と仮定する．もし，優れた潜在的関連集合

$PREL$を作ることができれば，マジック集合法が生成するga集合$S_IMP = MS \cup (IMP \cap PREL)$は$REL$に近づき，そして，仮定より，$|REL| << |IMP|$であるので，$S_IMP$は$IMP$に比べ大変小さくなり，故に，マジック集合法は，$PREL$の優れた程度に応じて，問題を効率的に解くことができる．

以上，マジック集合法を中心に，一般の問題を効率的に解くための従来の方法について説明した．

言葉の使い方について注意する．上のマジック集合法の説明の中で，$PREL$を作る，求める，という言い方をした．ところで，マジック集合法は$PREL$そのものを生成する（すなわち，$PREL$に属す全てのgaを生成する）のではなく，その代わりに，$PREL$を定義するもととなるga集合MSを生成している．問題例において，任意のga $p(X,Y)$をga $m_p(X)$に対応させるga写像をf（すなわち，$f(p(X,Y)) = m\_p(X)$）と記すと，$PREL$は$PREL = f^{-1}(MS)$と表されるので，正確には，マジック集合法は，$PREL$を定義するもととなるga集合MSと写像fを生成している．また，第5.3節の例に示すように，マジック集合法は，ルール集合P^{mg}を作ってそれによりga集合MSを定義するので，$PREL$を定義するもととなるルール集合P^{mg}と写像fを生成すると言うこともできる．本章では，マジック集合法に限らず，このように，$PREL$そのものを生成するのではなく，$PREL$を定義するもととなるものを生成する，作る，求めることを，簡単のため，単に，$PREL$を作る，求める，と言うので注意して欲しい．

次に緩和法について説明する．先に述べたように，本章では，一般の問題を効率的に解くための方法として緩和法を提案する．緩和法は，マジック集合法等，一般の問題を効率的に解くための従来の方法と同様，潜在的関連集合$PREL$を求めて，それを制約として使って，解の生成に関連しない不要なgaの生成をできるだけ防ぐことで，問題を効率的に解く．しかし，緩和法は，従来の方法とは異なり，$PREL$は関連集合RELの緩和であるという素直な考えに基づいて，初めRELを表すddbを作り，次にそのルール集合やファクト集合を簡略化して$PREL$を求める．

本章では，一般の問題を効率的に解くための従来の方法の中で最も適用範囲が広く，かつ，最も優れた$PREL$を作ることができる（すなわち，最も効率的である）マジックテンプレート法（ただし，アレクサンダーテンプレート法とHCT/R法もマジックテンプレート法と同じ能力をもつ）と緩和法を比較し，

1. マジックテンプレート法が解ける任意の問題に対し，緩和法が同じ$PREL$を作れる（他の$PREL$も作れる）こと，すなわち，任意の問題を緩和法がマジックテンプレート法と少なくとも同じ効率で解けること，

2. マジックテンプレート法が解けるある問題に対し，緩和法がマジックテンプレート法が作れない優れた$PREL$を作れること，すなわち，ある問題を緩和法がマジックテンプレート法より効率的に解けること，

3. マジックテンプレート法が解けないある問題を緩和法が効率的に解けること

を示す．1は，緩和法によりマジックテンプレート法を模倣できることを説明することにより示す．2と3は，そのような例を挙げることにより示す．2については例を三つ挙げる．その中には，緩和法がcyclic sipによる拘束の伝搬を反映した$PREL$を作る例や，簡単化されたファクト集合を反映した$PREL$を作る例が含まれる．また，3はdatalog問題ではないホーン問題に対する例を二つ挙げる．その中には，緩和法が，定数を含まない問い合わせアトム（例えば，$q(X,X)$?）に対して領域制限ルールによって定義される$PREL$を作る例が含まれる．1および3より，緩和法の適用範囲はマジックテンプレート法の適用範囲を真に包含している．すなわち，緩和法の適用範囲はdatalog問題クラスと一部のホーン問題であるが，3のために，ホーン問題部分がマジックテンプレート法より広くなっている．

なお，2および3を示す例，すなわち，緩和法がマジックテンプレート法より優れていることを示す例の中で用いる問題はいずれも，説明の簡単のため，ある単純な形をしており，そのため，これらの問題は逆計数法や拡張HaNa法の簡単な変形等を使って緩和法より効率的に解くことができる．しかし，これらの問題を少し変形することにより，緩和法のマジックテンプレート法に対する優位性を保持しつつ，逆計数法や拡張HaNa法の変形等が適用できない問題を作ることができる．これらの新しい問題に対しては，筆者らの知る限り，緩和法は従来のいかなる方法よりも効率的であると思われる．これらの新しい問題は，2，3の例の補足として，2，3の説明とは別の所に問題だけをまとめて示す．

最後に本章の以降の構成を述べる．第5.2節で緩和法を与える．第5.3節で2を示す．すなわち，緩和法がマジックテンプレート法が解ける問題をより効率的に解く例を与える．第5.4節で1を示す．すなわち，緩和法がマジックテンプレート法を模倣できることを説明する．第5.5節で3を示す．すなわち，緩和法がマジックテンプレート法が解けない問題を効率的に解く例を与える．第5.6節で2および3の例を補足する問題を与える．最後に，第5.7節でまとめる．

5.2 緩和法

緩和法を枠組みと緩和のための基本演算の二つに分けて説明する．なお，簡単のため，問題はdatalog問題であるとして説明する．

5.2.1 緩和法の枠組み

準備として幾つかの定義を与えた後に緩和法の枠組みを示す．

関連集合$REL(q(\boldsymbol{a},\boldsymbol{X})?,S)$に属しかつ述語名が$p(\in N(S))$である全てのgaの集合を$REL_p(q(\boldsymbol{a},\boldsymbol{X})?,S)(=\{p(\boldsymbol{b}) \mid p(\boldsymbol{b}) \in REL(q(\boldsymbol{a},\boldsymbol{X})?,S)\})$と記し，$IMP(S)$に属しかつ述語名が$p(\in N(S))$である全てのgaの集合を$IMP_p(S)(=\{p(\boldsymbol{b}) \mid p(\boldsymbol{b}) \in IMP(S)\})$と記す．次の定義5.3は，問題$(q(\boldsymbol{a},\boldsymbol{X})?,S)$の関連集合$REL(q(\boldsymbol{a},\boldsymbol{X})?,S)$を表すddb S^{REL}を定める．

定義 5.3 ddb $S=(P,D)$の問題$(q(\boldsymbol{a},\boldsymbol{X})?,S)$を考える．$S'=(P',D)$を別のddbとする．

$$\forall p \in N(S), \exists p' \in N(S') \quad REL_p(q(\boldsymbol{a},\boldsymbol{X})?,S)=\{p(\boldsymbol{b}) \mid p'(\boldsymbol{b}) \in IMP_{p'}(S')\} \tag{5.4}$$

が成立するとき，$p(\boldsymbol{Y})$に$p'(\boldsymbol{Y})$を対応させる写像をgと記し（すなわち，$p'(\boldsymbol{Y})=g(p(\boldsymbol{Y}))$），ddb S'を問題$(q(\boldsymbol{a},\boldsymbol{X})?,S)$の$g$の下での**関連ddb**（relevant ddb）と呼び，$S^{REL}=(P^{REL},D)$と記す．□

式(5.4)において，$IMP_{p'}(S')$を$IMP(S')$に変えても$REL_p(q(\boldsymbol{a},\boldsymbol{X})?,S)$は変わらないが，説明の簡単のため$IMP_{p'}(S')$と記す．

Sをddbとし，$HB(S)$に属すgaから成る全ての連言の集合を$HB(S)^*(=\{p(\boldsymbol{b}),\ldots,p(\boldsymbol{b})\wedge s(\boldsymbol{c}),\ldots,p(\boldsymbol{b})\wedge s(\boldsymbol{c})\wedge t(\boldsymbol{d}),\ldots \mid p(\boldsymbol{b}),s(\boldsymbol{c}),t(\boldsymbol{d}),\ldots \in HB(S)\})$と記し，同様に，$N(S)$に属す述語名から成る全ての連言の集合を$N(S)^*$と記す．また，$IMP_{p_1}(S)$に属すgaと，…，$IMP_{p_n}(S)$に属すgaとの連言の集合を$IMP_{p_1}(S)\otimes\cdots\otimes IMP_{p_n}(S)(=\{p_1(\boldsymbol{b}_1)\wedge\cdots\wedge p_n(\boldsymbol{b}_n) \mid p_i(\boldsymbol{b}_i)\in IMP_{p_i}(S),\ i=1,\ldots,n\})$と記す．次の定義5.4と5.5は，問題$(q(\boldsymbol{a},\boldsymbol{X})?,S)$の潜在的関連集合$PREL$を表すddb S^{RLXR}を，ddb S^{REL}の緩和として定める．定義5.4のddb S',S''は，各々，ddb S^{REL},S^{RLXR}に対応する．

定義 5.4 $S'=(P',D')$と$S''=(P'',D'')$をddbとする．$HB(S')$から$HB(S'')^*$への連言写像f：

$$f(p'(\boldsymbol{b}))=p_1''(\boldsymbol{b}_1)\wedge\cdots\wedge p_n''(\boldsymbol{b}_n) \tag{5.5}$$

が存在し，条件

$$\begin{aligned}&\forall p' \in N(S'), \exists p_1'', p_2'', \ldots, p_n'' \in N(S'')\\ &f(IMP_{p'}(S'))(=\{f(p'(\boldsymbol{b})) \mid p'(\boldsymbol{b}) \in IMP_{p'}(S')\})\\ &\quad \subseteq IMP_{p_1''}(S'')\otimes IMP_{p_2''}(S'')\otimes\cdots\otimes IMP_{p_n''}(S'')\end{aligned} \tag{5.6}$$

が成立するとき，S''はfの下でS'の**緩和**(relaxation)であると言う．また，このようなddb S''および写像fを作ることを，S'を**緩和する**と言う．□

式(5.5)のfにおいて，定数ベクトル$\boldsymbol{b}$および$\boldsymbol{b}_i$の次元は対応する述語p'およびp_i''の引数の数と一致しているものとする．引数ベクトル$\boldsymbol{Y}=(Y_1,\ldots,Y_m)$に含まれる変数の集合を$\langle\boldsymbol{Y}\rangle$（$=\{Y_1,\ldots,Y_m\}$）と記す．多くの緩和法（すなわち，第5.2.2節の定数グループ化を用いない緩和法）において，p_i''の引数集合$\langle\boldsymbol{Y}_i\rangle$は$p'$の引数集合$\langle\boldsymbol{Y}\rangle$の部分集合であり，$\boldsymbol{b}_i$は$\boldsymbol{b}$の対応する引数の要素を選んだものになっている．このとき，fは$N(S')$から$N(S'')^*$への写像

$$f(p'(\boldsymbol{Y}))=p_1''(\boldsymbol{Y}_1)\wedge\cdots\wedge p_n''(\boldsymbol{Y}_n) \tag{5.7}$$

と考えることができる．但し，第5.2.2節の定数グループ化を用いる緩和法では，定数のグループ化を表す$C(S')$から$C(S'')$への定数写像をf_cと記すと，$\boldsymbol{b}_i$の各要素の値は$\boldsymbol{b}$の対応する要素の値のf_cによる像となっている．すなわち，$b_{i_1},\ldots,b_{i_m}$を$\boldsymbol{b}$のある要素として，$\boldsymbol{b}_i=(b_{i1},\ldots,b_{im})=(f_c(b_{i_1}),\ldots,f_c(b_{i_m}))$である．このとき，$\langle\boldsymbol{Y}_i\rangle$を$p'$の引数集合$\langle\boldsymbol{Y}\rangle$の部分集合として，$f$は写像

$$f(p'(\boldsymbol{Y}))=p_1''(f_c(\boldsymbol{Y}_1))\wedge\cdots\wedge p_n''(f_c(\boldsymbol{Y}_n)) \tag{5.8}$$

と考えることができる．ここで，$\boldsymbol{Y}_i=(Y_{i1},\ldots,Y_{im})$として，$(f_c(Y_{i1}),\ldots,f_c(Y_{im}))$を$f_c(\boldsymbol{Y}_i)$と記した．

定義 5.5 ddb $S^{REL}=(P^{REL},D)$を問題$(q(\boldsymbol{a},\boldsymbol{X})?,S)$の$g$の下での関連ddbとし，ddb S''をfの下でのS^{REL}の緩和とする．このとき，S''を問題$(q(\boldsymbol{a},\boldsymbol{X})?,S)$の$f\cdot g$の下での**緩和関連ddb**（relaxed relevant ddb）と呼び，$S^{RLXR}=(P^{RLXR},D^{RLXR})$と記す．□

S^{REL},g,S^{RLXR},fを定義5.5のそれらとする．このとき，式(5.4)および(5.6)より

$$REL(q(\boldsymbol{a},\boldsymbol{X})?,S)=\bigcup_{p\in N(S)}\{p(\boldsymbol{b}) \mid g(p(\boldsymbol{b})) \in IMP_{p'}(S^{REL})\}$$

step1: 問題 $(q(\boldsymbol{a},\boldsymbol{X})?,S)$ の関連ddb $S^{REL}=(P^{REL},D)$ および写像 g を作る．

step2: S^{REL} を緩和し，緩和関連ddb $S^{RLXR}=(P^{RLXR},D^{RLXR})$ および写像 f を作る．

step3: S^{RLXR} および $f\cdot g$ が定める潜在的関連集合 $PREL$（式(5.9)）を制約として元のddb S に付加することにより新しいddb（**変形ddb**（modified ddb）と呼び，S^{MDF} と記す）を作る．すなわち，S の各ルール $r(\in P)$

$p(\boldsymbol{Y})\leftarrow t_1(\boldsymbol{Z}_1),t_2(\boldsymbol{Z}_2),\ldots,t_h(\boldsymbol{Z}_h).$

に対し，頭部述語 $p(\boldsymbol{Y})$ の $f\cdot g$ による像 $f(g(p(\boldsymbol{Y})))=p_1''(f_c(\boldsymbol{Y}_1))\wedge\cdots\wedge p_n''(f_c(\boldsymbol{Y}_n))$（式(5.8)）を r の本体に付加してルール r'''

$p(\boldsymbol{Y})\leftarrow p_1''(f_c(\boldsymbol{Y}_1)),\ldots,p_1''(f_c(\boldsymbol{Y}_n)),t_1(\boldsymbol{Z}_1),t_2(\boldsymbol{Z}_2),\ldots,t_h(\boldsymbol{Z}_h).$

を作る．P''' をそのようなルール r''' の集合として，変形ddb $S^{MDF}=(P'''\cup P^{RLXR},D\cup D^{RLXR})$ を作る．

step4: 変形ddb S^{MDF} と問い合わせアトム $q(\boldsymbol{a},\boldsymbol{X})?$ よりなる問題 $(q(\boldsymbol{a},\boldsymbol{X})?,S^{MDF})$ をセミナイーブ法を使って解いて解集合 ANS を得る．

図5.2: 緩和法の枠組み

$$\subseteq \bigcup_{p\in N(S)}\{p(\boldsymbol{b})\mid p(\boldsymbol{b})\in HB(S),\ f(g(p(\boldsymbol{b})))\in IMP_{p_1''}(S^{RLXR})\otimes\cdots\otimes IMP_{p_n''}(S^{RLXR})\}$$

が成立するので，ga集合

$$PREL=\bigcup_{p\in N(S)}\{p(\boldsymbol{b})\mid p(\boldsymbol{b})\in HB(S),\ f(g(p(\boldsymbol{b})))\in IMP_{p_1''}(S^{RLXR})\otimes\cdots\otimes IMP_{p_n''}(S^{RLXR})\}\tag{5.9}$$

は第5.1節の緩和条件(5.3)を満足する．緩和法は，この $PREL$ を解の生成に関連しない不要なga $(\in IMP(S))$ の生成を防ぐための制約として使う．

緩和法の枠組みを図5.2に与える．step4の新しい問題 $(q(\boldsymbol{a},\boldsymbol{X})?,S^{MDF})$ の解集合 ANS は元の問題 $(q(\boldsymbol{a},\boldsymbol{X})?,S)$ の解集合に等しい．step1において関連ddb S^{REL} および写像 g を作る方法については，第5.3節の例と第5.4節の例の中で二つの方法を与える．step2においてddb S^{REL} を緩和してddb S^{RLXR} および写像 f を作る方法については，第5.2.2節で基本となる五つの緩和演算を与えて説明する．緩和法の例は第5.3節以降に与える．

5.2.2 緩和のための基本演算

S^{REL} を緩和し S^{RLXR} と f を作るための基本演算を五つ与える．これらの基本演算は組み合わせて用いることができる．演算をどのように選択し適用しても，(i) S^{REL},S^{RLXR},f は式(5.6)の条件を満足する．故に，緩和法は解を正しく求めることができる．効率を高めるためには，(ii) セミナイーブ法が S^{RLXR} をボトムアップ評価する際の時間量，領域量が小さい，(iii) S^{RLXR} と f と g が定める $PREL$ が REL をできるだけうまく近似する，ことが重要である．そのためには，S^{REL} のもつ重要な拘束を残しつつ S^{REL} を簡略化するように，演算を適切に選択し適用することが重要である．以下の(1)から(5)の説明では，元になるddbを $S'=(P',D')$，演算を適用した結果を $S''=(P'',D'')$ と記す．

(1) **述語消去**（elimination of predicate）

P' の一つのルール r' に注目し，その本体にあるあるアトムを消去することによりルール r'' を作り，r' と置き換える．$P''=P'-\{r'\}\cup\{r''\}$，$D''=D'$ である．f は，任意のIDB述語 $p(\in N(S'))$ について，恒等写像 $f(p(\boldsymbol{Y}))=p(\boldsymbol{Y})$ とする．この演算を**述語消去**と呼ぶ．ルール $r'(\in P')$ の本体に含まれる異なる変数の数を $deg(r')$ と記すと，粗い見積もりでは，r' をボトムアップ評価する最悪時間量は $O(|C(S')|^{deg(r')})$ である．従って，時間量を減らすためには，$deg(r')$ または $|C(S')|$ を減少させることが有効である．ルール r' の本体からある変数を含む全てのアトムが消去されるように述語消去演算を複数回適用してルール r''' を作ると，$deg(r''')<deg(r')$ となるので，時間量の小さなルール r''' を得ることができる．

(2) **本体述語分割**（decomposition of body predicate）

P' の一つのルール

r' : $\quad q\leftarrow p(\boldsymbol{X}),s,t,\ldots,u.$

に注目し，その本体にあるあるIDBアトム，例えば $p(\boldsymbol{X})$ を

$$p(\boldsymbol{X})\rightarrow p_1(\boldsymbol{X}_1)\wedge\cdots\wedge p_k(\boldsymbol{X}_k),\quad \langle\boldsymbol{X}_1\rangle,\ldots,\langle\boldsymbol{X}_k\rangle\subset\langle\boldsymbol{X}\rangle,\ k\geq 1$$

のように分割し（ここで，$\langle\boldsymbol{X}_i\rangle$ と $\langle\boldsymbol{X}_j\rangle$ は同じ変数を重複して含んでもよい），r' の $p(\boldsymbol{X})$ を $p_1(\boldsymbol{X}_1)\wedge\cdots\wedge p_k(\boldsymbol{X}_k)$ で置き換えてルール r'' を作る．更に，$p_1(\boldsymbol{X}_1),\ldots,p_k(\boldsymbol{X}_k)$ を定義するルール $r_1'',\ldots,r_k''$ を加える．

r'' : $\quad q\leftarrow p_1(\boldsymbol{X}_1),\ldots,p_k(\boldsymbol{X}_k),s,t,\ldots,u.$

r_1'' : $\quad p_1(\boldsymbol{X}_1)\leftarrow p(\boldsymbol{X}).$

$$\vdots$$

r''_k : $\quad p_k(\boldsymbol{X}_k) \leftarrow p(\boldsymbol{X})$.

この場合もfは恒等写像とする．この演算を**本体述語分割**と呼ぶ．特に，$k=1$のとき，変数集合$\langle \boldsymbol{X} \rangle$が$\langle \boldsymbol{X}_1 \rangle$に縮小されるので，$k=1$のときの本体述語分割を**本体引数消去**（elimination of body argument）と呼ぶ．

例えば，ルールr'の本体にあるある変数Xに着目し，Xを含む全てのアトム$(p(\boldsymbol{X}), s(\boldsymbol{Y})$とする$)$が変数$X$を消去してできるアトム$p_1(\boldsymbol{X}_1), s_1(\boldsymbol{Y}_1)$に置き換わるように，$r'$に本体引数消去を複数回適用してルール$r'''$を作る．このとき，$deg(r''') < deg(r')$であるので，時間量の小さなルール$r'''$を得ることができる．

また，アトム$p(\boldsymbol{X}), s(\boldsymbol{Y})$が変数$X$を含まない部分$(p_1(\boldsymbol{X}_1), s_1(\boldsymbol{Y}_1)$と記す$)$と変数$X$を含む部分$(p_2(\boldsymbol{X}_2), s_2(\boldsymbol{Y}_2)$と記す$)$との連言$p_1(\boldsymbol{X}_1) \wedge p_2(\boldsymbol{X}_2), s_1(\boldsymbol{Y}_1) \wedge s_2(\boldsymbol{Y}_2)$に置き換わるように，$r'$に本体述語分割演算$(k=2)$を複数回適用して$p_1(\boldsymbol{X}_1), p_2(\boldsymbol{X}_2), s_1(\boldsymbol{Y}_1), s_2(\boldsymbol{Y}_2)$を含むルール$r'''$を作り，更に，$r'''$が$p_1(\boldsymbol{X}_1), s_1(\boldsymbol{Y}_1)$を含むルール$r'''_1$と$p_2(\boldsymbol{X}_2), s_2(\boldsymbol{Y}_2)$を含むルール$r'''_2$に分割されるように，$r'''$に次のルール分割演算(3)を適用してルール$r'''_1, r'''_2$を作る．こうすれば，$deg(r'''_1), deg(r'''_2) < deg(r')$となるので，変数$X$を$r'''_2$に残しつつ，時間量の小さなルール$r'''_1, r'''_2$を得ることができる．

本体引数消去は述語消去より，また，本体述語分割$(k \geq 2)$は本体引数消去より，細かなルール書き換えが可能である．

(3) **ルール分割**（decomposition of rule）

P'のルール

r' : $\quad p(\boldsymbol{X}) \leftarrow q_1(\boldsymbol{X}_1), q_2(\boldsymbol{X}_2), \ldots, q_n(\boldsymbol{X}_n)$.

は本体に多くの変数，多くのアトムを含むとする．r'の本体のアトムを二つのグループ$\{q_1(\boldsymbol{X}_1), \ldots, q_l(\boldsymbol{X}_l)\}$と$\{q_h(\boldsymbol{X}_h), \ldots, q_n(\boldsymbol{X}_n)\}$に分ける．但し，$\{q_1(\boldsymbol{X}_1), \ldots, q_l(\boldsymbol{X}_l)\} \cup \{q_h(\boldsymbol{X}_h), \ldots, q_n(\boldsymbol{X}_n)\} = \{q_1(\boldsymbol{X}_1), \ldots, q_n(\boldsymbol{X}_n)\}$であるが，同じアトムが二つのグループに重複して含まれてもよい．グループ$\{q_1(\boldsymbol{X}_1), \ldots, q_l(\boldsymbol{X}_l)\}$に含まれ，かつ，グループ$\{q_h(\boldsymbol{X}_h), \ldots, q_n(\boldsymbol{X}_n)\}$または頭部$p(\boldsymbol{X})$に含まれる変数よりなる引数ベクトルを$\boldsymbol{X}'$と記す．ルール$r'$を次の二つのルール$r''_1$と$r''_2$に分割して置き換える．

r''_1 : $\quad p'(\boldsymbol{X}') \leftarrow q_1(\boldsymbol{X}_1), \ldots, q_l(\boldsymbol{X}_l)$.

r''_2 : $\quad p(\boldsymbol{X}) \leftarrow p'(\boldsymbol{X}'), q_h(\boldsymbol{X}_h), \ldots, q_n(\boldsymbol{X}_n)$.

この場合もfは恒等写像とする．この演算を**ルール分割**と呼ぶ．$deg(r''_1), deg(r''_2) < deg(r')$となるようにルール分割演算を適用すれば，時間量の小さなルールr''_1, r''_2を得ることができる．

述語p'の引数の数を$deg(p')$と記すとき，粗い見積もりでは，$IMP_{p'}(S'')$を蓄えるための最悪領域量は$O(|C(S'')|^{deg(p')})$となる．この領域量が大きすぎる場合は，p'に次に説明する頭部引数消去演算(4)を適用し$deg(p')$を減らすか，または，$D''(=D')$に後述の定数グループ化演算(5)を適用し$|C(S'')|$を減らす必要がある．

なお，本ルール分割は第5.1節の初めに述べたマジック集合法等のルール分割を一般化したものであり，緩和法ではS^{REL}を緩和してS^{RLXR}を作る際に利用している．以後，ルール分割は本ルール分割を示すものとする．

(4) **頭部述語分割**（decomposition of head predicate）

あるIDB述語$p(\in N(S'))$を頭部または本体にもつ全てのルールを$r'_1, \ldots, r'_h(\in P')$と記す．述語$p$の分割

$$p(\boldsymbol{X}) \rightarrow p_1(\boldsymbol{X}_1) \wedge \cdots \wedge p_k(\boldsymbol{X}_k), \quad \langle \boldsymbol{X}_1 \rangle, \ldots, \langle \boldsymbol{X}_k \rangle \subset \langle \boldsymbol{X} \rangle,\ k \geq 1$$

を定め（ここで，$\langle \boldsymbol{X}_i \rangle$と$\langle \boldsymbol{X}_j \rangle$は同じ変数を重複して含んでもよい），各ルール

r'_i : $\quad p \leftarrow p, q_1, \ldots, q_l$.

に対し，その本体に含まれる全てのpを$p_1 \wedge \cdots \wedge p_k$で置き換え，各$p_j, j=1, \ldots, k,$を頭部とするルール$r''_{i1}, \ldots, r''_{ik}$を作る．

r''_{i1} : $\quad p_1 : -p_1, \ldots, p_k, q_1, \ldots, q_l$.

$$\vdots$$

r''_{ik} : $\quad p_k : -p_1, \ldots, p_k, q_1, \ldots, q_l$.

（ルールr'_iがpを頭部に持たないときは，本体のpのみが$p_1 \wedge \cdots \wedge p_k$と置き換えられるので，ルールは一つだけできる．）写像fは，述語pについては$f(p(\boldsymbol{X})) = p_1(\boldsymbol{X}_1) \wedge \cdots \wedge p_k(\boldsymbol{X}_k)$，$p$以外の述語については恒等写像とする．この演算を**頭部述語分割**と呼ぶ．また，特に，$k=1$のときの頭部述語分割を**頭部引数消去**（elimination of head argument）と呼ぶ．$deg(p_j) < deg(p), j=1, \ldots, k,$であるので，$r''_{ij}$の領域量は一般に小さくなる．ルール$r''_{i'j'}$が頭部$p_{j'}(\boldsymbol{X}_{j'})$の生成にはあまり関係のないアトムを本体に含む場合，通常，そのルール$r''_{i'j'}$に更に述語消去演算(1)を適用して不要なアトムを消去する．

(5) **定数グループ化**（grouping of constant）

$S' = (P', D')$に現れる定数を幾つかのグループ（$c_1, \ldots, c_m$と記す）に分ける．このグループ化を表すmany-to-one写像をf_c： $C(S') \to \{c_1, \ldots, c_m\}$として，$S'$のルールおよびファクトに現れる各定数$b$を$f_c(b)$に置き換えることにより$S'' = (P'', D'')$を作る．すなわち，$S''$のファクト集合$D''$を

$$D'' \quad = \quad \{A'(f_c(b_1), \ldots, f_c(b_h)) \mid A \text{は} P' \text{のEDB述語, かつ,} A(b_1, \ldots, b_h) \in D'\}$$

とする．また，EDB述語または定数を含むP'の全てのルールを$r'_i, i = 1, 2, \ldots, n$, として，それらに現れる各EDB述語$A$および各定数$b$を$A', f_c(b)$と置き換えてルール$r''_i$を作り，できた$r''_i$を$r'_i$と置き換えて$S''$のルール集合$P''$とする．写像$f$は任意の述語$p$について

$$f(p(X_1, \ldots, X_m)) \quad = \quad p(f_c(X_1), \ldots, f_c(X_m)) \tag{5.10}$$

とする．この演算を**定数グループ化**と呼ぶ．$|C(S'')| < |C(S')|$であるので，領域量および時間量の小さなS''を得ることができる．

上記の(1)から(5)の説明は，第5.2節の初めに述べたように，問題がdatalog問題である，すなわち，ddb S'がdatalog ddbであるとして説明した．問題がdatalog問題ではないホーン問題である場合，すなわち，ddb S'がdatalog ddbではないホーンddbである場合，(1)から(4)の緩和基本演算は，このような場合であっても，使用することができる．しかし，定数グループ化(5)については，S'が算術述語を含む場合，新しい定数$c_1, \ldots, c_m$への算術述語の適用は一般に不可能であるので，S'が算術述語を含む場合は定数グループ化は使用できない．

述語消去，頭部引数消去，簡単なルール分割は従来の多くの問い合わせ処理法で使われている．頭部述語分割($k \geq 2$)は[NRSU89a]で使われており，そこでは，P'とP''が等価になる場合が議論されている．

5.3 緩和法が従来の方法より効率的な例

マジックテンプレート法が解ける問題を緩和法がより効率的に解く例を三つ与える．

5.3.1 例5.1

ddb $S1 = (P1, D1)$のルール集合を

$$P1: \quad p(X, Y, Z) \leftarrow A(X, Y, Z). \tag{5.11}$$

$$p(X, Y, Z) \leftarrow B(X', X), C(Y', Y), D(Z', Z), p(X', Y', Z'). \tag{5.12}$$

$$s(X, Y, Z) \leftarrow E(X, Y, Z). \tag{5.13}$$

$$s(X, Y, Z) \leftarrow F(X', X), G(Y', Y), H(Z', Z), s(X', Y', Z'). \tag{5.14}$$

$$q(X, Y, Z, Z') \leftarrow p(X, Y, Z), s(X, Y, Z'). \tag{5.15}$$

とし（ファクト集合$D1$は省略），問い合わせアトムを$q(a, Y, Z, Z')?$とする．問題$(q(a, Y, Z, Z')?, S1)$（問題1と呼ぶ）について考える．

問題1は次のように解釈できる．3次元空間xyzの格子点上を移動可能な乗り物V_pとV_sがある．述語$B(X', X)$はV_pがx座標X'からXへ直接移動できることを，述語$C(Y', Y)$と$D(Z', Z)$は，各々，y座標とz座標について同様なことを示す．また，述語$F(X', X), G(Y', Y), H(Z', Z)$は，各々，$V_s$の$x, y, z$座標について同様なことを示す．述語$p(X, Y, Z)$（$s(X, Y, Z)$）は$V_p$（$V_s$）が3次元座標$(X, Y, Z)$に到達可能であることを表し，述語$q(X, Y, Z, Z')$は，$V_p$と$V_s$が共に到達可能な$xy$座標$(X, Y)$と，そのときの$V_p$の$z$座標$Z$と$V_s$の$z$座標$Z'$との組$(X, Y, Z, Z')$を表す．問い合わせアトム$q(a, Y, Z, Z')?$は，それらの$q(X, Y, Z, Z')$の中で特に$x$座標の値が$a$であるものを選択する．

初めに緩和法による解法例を示す．

(1) **step1**

ルール集合$P1$と問い合わせアトム$q(a, Y, Z, Z')?$より次のルール集合$P1^{REL}$: (5.16), ..., (5.25), を作る．

$$o\_p(X, Y, Z) \leftarrow A(X, Y, Z). \quad (\text{(5.11) より}) \tag{5.16}$$

$$o\_p(X, Y, Z) \leftarrow B(X', X), C(Y', Y), D(Z', Z), o\_p(X', Y', Z'). \quad (\text{(5.12) より}) \tag{5.17}$$

$$o\_s(X, Y, Z) \leftarrow E(X, Y, Z). \quad (\text{(5.13) より}) \tag{5.18}$$

$$o\_s(X, Y, Z) \leftarrow F(X', X), G(Y', Y), H(Z', Z), o\_s(X', Y', Z'). \quad (\text{(5.14) より}) \tag{5.19}$$

$$o\_q(X,Y,Z,Z') \leftarrow o\_p(X,Y,Z), o\_s(X,Y,Z'). \quad ((5.15)\text{より}) \tag{5.20}$$

$$r\_q(a,Y,Z,Z') \leftarrow o\_q(a,Y,Z,Z'). \quad (\text{問い合わせアトムより}) \tag{5.21}$$

$$r\_p(X,Y,Z) \leftarrow r\_q(X,Y,Z,Z'), o\_p(X,Y,Z), o\_s(X,Y,Z'). \quad ((5.15)\text{より}) \tag{5.22}$$

$$r\_p(X',Y',Z') \leftarrow r\_p(X,Y,Z), B(X',X), C(Y',Y), D(Z',Z), o\_p(X',Y',Z'). \quad ((5.12)\text{より}) \tag{5.23}$$

$$r\_s(X,Y,Z') \leftarrow r\_q(X,Y,Z,Z'), o\_p(X,Y,Z), o\_s(X,Y,Z'). \quad ((5.15)\text{より}) \tag{5.24}$$

$$r\_s(X',Y',Z') \leftarrow r\_s(X,Y,Z), F(X',X), G(Y',Y), H(Z',Z), o\_s(X',Y',Z'). \quad ((5.14)\text{より}) \tag{5.25}$$

ルール(5.16),...,(5.20)は，述語名p,s,qがo_p,o_s,o_qに変更されている点を除き，元のルール(5.11), ..., (5.15)に等しい．従って，ルール(5.16), ..., (5.20)をボトムアップ評価すると，$IMP_{o\_p}, IMP_{o\_s}, IMP_{o\_q}$（すなわち，$IMP_p, IMP_s, IMP_q$）が生成される．ルール(5.21), ..., (5.25)はこれらのgaを問い合わせアトム$q(a,Y,Z,Z')?$に関連づけるものであって，r_q, r_p, r_sはその結果を表す述語である．まず，ルール(5.21)の本体のアトム$o_q(a,Y,Z,Z')$は解$q(a,Y,Z,Z')$を与えるから，頭部アトム$r_q(a,Y,Z,Z')$は解の関連gaを与える．(5.22)の頭部アトム$r_p(X,Y,Z)$は，ルール(5.15)を使って$q(a,Y,Z,Z')$の関連gaを生成する際に使われる$p(X,Y,Z)$の関連gaを与え，(5.23)の頭部アトム$r_p(X',Y',Z')$は，ルール(5.12)を使って$p(X,Y,Z)$の関連gaを生成する際に使われる$p(X',Y',Z')$の関連gaを与える．ルール(5.24), (5.25)の頭部アトムr_sも，r_pと同様に，sについての関連gaを与える．すなわち，$S1^{REL} = (P1^{REL}, D1) = (\{(5.16), \ldots, (5.25)\}, D1)$は写像$g1$:

$$p \rightarrow r\_p, \quad s \rightarrow r\_s, \quad q \rightarrow r\_q, \quad (\text{引数は省略})$$

の下で問題$(q(a,Y,Z,Z')?, S1)$の関連ddbである．

(2) **step2**

緩和関連ddbを作る一つの方法として，ルール集合$P1^{REL}$の各述語からz座標に対応する引数を消去し（すなわち，第5.2.2節の頭部引数消去），次のルール集合$P1^{RLXR}$：(5.26),...,(5.35),を作る．

$$ro\_p(X,Y) \leftarrow A(X,Y,Z). \quad ((5.16)\text{より}) \tag{5.26}$$

$$ro\_p(X,Y) \leftarrow B(X',X), C(Y',Y), ro\_p(X',Y'). \quad ((5.17)\text{より}) \tag{5.27}$$

$$ro\_s(X,Y) \leftarrow E(X,Y,Z). \quad ((5.18)\text{より}) \tag{5.28}$$

$$ro\_s(X,Y) \leftarrow F(X',X), G(Y',Y), ro\_s(X',Y'). \quad ((5.19)\text{より}) \tag{5.29}$$

$$ro\_q(X,Y) \leftarrow ro\_p(X,Y), ro\_s(X,Y). \quad ((5.20)\text{より}) \tag{5.30}$$

$$rr\_q(a,Y) \leftarrow ro\_q(a,Y). \quad ((5.21)\text{より}) \tag{5.31}$$

$$rr\_p(X,Y) \leftarrow rr\_q(X,Y), ro\_p(X,Y), ro\_s(X,Y). \quad ((5.22)\text{より}) \tag{5.32}$$

$$rr\_p(X',Y') \leftarrow rr\_p(X,Y), B(X',X), C(Y',Y), ro\_p(X',Y'). \quad ((5.23)\text{より}) \tag{5.33}$$

$$rr\_s(X,Y) \leftarrow rr\_q(X,Y), ro\_p(X,Y), ro\_s(X,Y). \quad ((5.24)\text{より}) \tag{5.34}$$

$$rr\_s(X',Y') \leftarrow rr\_s(X,Y), F(X',X), G(Y',Y), ro\_s(X',Y'). \quad ((5.25)\text{より}) \tag{5.35}$$

ここで，EDB述語DとHのアトムはz座標に対応する引数のみを持つので，それらのアトムを消去した（すなわち，第5.2.2節の述語消去）．

$S1^{RLXR} = (P1^{RLXR}, D1) = (\{(5.26), \ldots, (5.35)\}, D1)$は写像$f1$:

$$r\_p(X,Y,Z) \rightarrow rr\_p(X,Y), \quad r\_s(X,Y,Z) \rightarrow rr\_s(X,Y),$$

$$r\_q(X,Y,Z,Z') \rightarrow rr\_q(X,Y),$$

の下で$S1^{REL}$の緩和になる（述語o_p, o_s, o_qに関する写像は省略）．

(3) **step3**

ルール集合$P1$：(5.11),...,(5.15),の各ルールの本体に，頭部アトム$p(X,Y,Z), s(X,Y,Z), q(X,Y,Z,Z')$についての制約$f1(g1(p(X,Y,Z))) = rr_p(X,Y)$, $f1(g1(s(X,Y,Z))) = rr\_s(X,Y)$, $f1(g1(q(X,Y,Z,Z'))) = rr_q(X,Y)$を各々加えることにより，次のルール集合$P1'''$：(5.36),...,(5.40),を作る．

$$p(X,Y,Z) \leftarrow rr\_p(X,Y), A(X,Y,Z). \tag{5.36}$$

$$p(X,Y,Z) \leftarrow rr\_p(X,Y), B(X',X), C(Y',Y), D(Z',Z), p(X',Y',Z'). \tag{5.37}$$

$$s(X,Y,Z) \leftarrow rr\_s(X,Y), E(X,Y,Z). \tag{5.38}$$

$$s(X,Y,Z) \leftarrow rr\_s(X,Y), F(X',X), G(Y',Y), H(Z',Z), s(X',Y',Z'). \tag{5.39}$$

$$q(X,Y,Z,Z') \leftarrow rr\_q(X,Y), p(X,Y,Z), s(X,Y,Z'). \tag{5.40}$$

以上の結果，変形ddb $S1^{MDF} = (P1''' \cup P1^{RLXR}, D1)$ が得られる．

(4) **step4**

例えば，ファクト集合 $D1$ を

$$\begin{aligned} D1 \;=\; & \{A(1,100,1), E(1,91,1)\} \\ & \cup \{B(i,j), C(j,i), D(i,j), F(i,j), G(i+90, j+90), H(i,j) \mid \\ & \quad 1 \le i \le j \le 100, (j = i+1 \text{または} j = i), i, j : \text{整数}\} \end{aligned} \tag{5.41}$$

とし，問い合わせアトム中の定数 a を $a = 80$ とする．ddb $S1^{MDF}$ をセミナイーブ法を使ってボトムアップ評価すると，ga集合

$$\begin{aligned} IMP(S1^{MDF}) = & \{ro\_p(x,y) \mid 1 \le x \le 100, 1 \le y \le 100\} \\ & \cup \{ro\_s(x,y) \mid 1 \le x \le 100, 91 \le y \le 190\} \\ & \cup \{ro\_q(x,y) \mid 1 \le x \le 100, 91 \le y \le 100\} \\ & \cup \{rr\_q(x,y) \mid x = 80, 91 \le y \le 100\} \\ & \cup \{rr\_p(x,y) \mid 1 \le x \le 80, 91 \le y \le 100\} \\ & \cup \{rr\_s(x,y) \mid 1 \le x \le 80, 91 \le y \le 100\} \\ & \cup \{p(x,y,z) \mid 1 \le x \le 80, 91 \le y \le 100, 1 \le z \le 100\} \\ & \cup \{s(x,y,z) \mid 1 \le x \le 80, 91 \le y \le 100, 1 \le z \le 100\} \\ & \cup \{q(x,y,z,z') \mid 1 \le x \le 80, 91 \le y \le 100, 1 \le z, z' \le 100\} \end{aligned} \tag{5.42}$$

が生成される（ここで，各gaの引数の値はいずれも整数とする）．$IMP(S1^{MDF})$ の中から解を選択して，解集合

$$ANS1 \;=\; \{q(x,y,z,z') \mid x = 80, 91 \le y \le 100, 1 \le z, z' \le 100\} \tag{5.43}$$

を得る．

$IMP(S1^{MDF})$（式(5.42)）の中の述語 p のgaの集合を $IMP_p(S1^{MDF})$ と記し，他の述語のgaの集合も同様に記す．$IMP(S1^{MDF})$ のうち $IMP_{ro\_p}(S1^{MDF}) \cup IMP_{ro\_s}(S1^{MDF}) \cup IMP_{ro\_q}(S1^{MDF}) \cup IMP_{rr\_q}(S1^{MDF}) \cup IMP_{rr\_p}(S1^{MDF}) \cup IMP_{rr\_s}(S1^{MDF})$ は潜在的関連集合 $PREL1$ を求めるために生成したga集合である．$PREL1$ を陽に表すと

$$PREL1 = \{p(x,y,z) \mid rr\_p(x,y) \in IMP_{rr\_p}(S1^{MDF}), z \text{は任意の整数}\}$$

$$\begin{aligned} & \cup \{s(x,y,z) \mid rr\_s(x,y) \in IMP_{rr\_s}(S1^{MDF}), z \text{は任意の整数}\} \\ & \cup \{q(x,y,z,z') \mid rr\_q(x,y) \in IMP_{rr\_q}(S1^{MDF}), z \text{と} z' \text{は任意の整数}\} \end{aligned} \tag{5.44}$$

となる．問題 $(q(a,Y,Z,Z')?, S1)$ の xy 座標のみを残し z 座標を消去した小規模な問題 $(q'(a,Y)?, S1')$ を考えると，$IMP_{rr\_q}(S1^{MDF}) \cup IMP_{rr\_p}(S1^{MDF}) \cup IMP_{rr\_s}(S1^{MDF})$ はこの問題の関連集合である（$REL(q'(a,Y)?, S1') = IMP_{rr\_q}(S1^{MDF}) \cup IMP_{rr\_p}(S1^{MDF}) \cup IMP_{rr\_s}(S1^{MDF})$）ので，$PREL1$ は，第1引数と第2引数が小規模問題の関連集合 $REL(q'(a,Y)?, S1')$ に含まれるような p ga，s ga，q gaの集合になっている．$IMP(S1^{MDF})$ のうち残りの $IMP_p(S1^{MDF}) \cup IMP_s(S1^{MDF}) \cup IMP_q(S1^{MDF})$ は，$PREL1$ を制約として使って，問題1のddb $S1$ より生成可能なgaのうち $PREL1$ に含まれるようなgaのみを生成した結果できたga集合であり，ddb $S1$ の意味

$$\begin{aligned} IMP(S1) \;=\; & \{p(x,y,z) \mid 1 \le x, y, z \le 100\} \\ & \cup \{s(x,y,z) \mid 1 \le x, z \le 100, 91 \le y \le 190\} \\ & \cup \{q(x,y,z,z') \mid 1 \le x, z, z' \le 100, 91 \le y \le 190\} \end{aligned}$$

と潜在的関連集合 $PREL1$ の共通部分になっている（$IMP_p(S1^{MDF}) \cup IMP_s(S1^{MDF}) \cup IMP_q(S1^{MDF}) = IMP(S1) \cap PREL1$）．

$|IMP_{ro\_p}(S1^{MDF}) \cup IMP_{ro\_s}(S1^{MDF}) \cup IMP_{ro\_q}(S1^{MDF}) \cup IMP_{rr\_q}(S1^{MDF}) \cup IMP_{rr\_p}(S1^{MDF}) \cup IMP_{rr\_s}(S1^{MDF})| << |IMP(S1)|$，かつ，$|IMP_p(S1^{MDF}) \cup IMP_s(S1^{MDF}) \cup IMP_q(S1^{MDF})| << |IMP(S1)|$ であるので，問題1をセミナイーブ法を使って解いた場合と比べ，緩和法は問題1をより効率的に解いている．

次に，同じ問題1に対するマジックテンプレート法の解法例を示す．問題1のルール(5.15)には二つのacyclic sip:

$$\begin{aligned} \text{sip1} \;=\; & \{\{q_h(X)\} \rightarrow_{\{X\}} p(X,Y,Z),\ \{q_h(X), p(X,Y,Z)\} \rightarrow_{\{X,Y\}} s(X,Y,Z')\}, \\ \text{sip2} \;=\; & \{\{q_h(X)\} \rightarrow_{\{X\}} s(X,Y,Z'),\ \{q_h(X), s(X,Y,Z')\} \rightarrow_{\{X,Y\}} p(X,Y,Z)\} \end{aligned}$$

がある．ここで，アトム $q_h(X)$ は，問い合わせアトム $q(a,Y,Z,Z')?$ において定数に固定されている引数（すなわち，第1引数）のみに限定されたルール(5.15)の頭部アトム $q(X,Y,Z,Z')$ である．例えば，sip1は，初め，頭部アトム $q(X,Y,Z,Z')$ の第1引数 X についての拘束を本体アトム $p(X,Y,Z)$ の第1引数 X に伝搬し（$\{q_h(X)\} \rightarrow_{\{X\}} p(X,Y,Z)$），次に，本体アトム $p(X,Y,Z)$ の第1引数 X および第2引数 Y についての拘束を本体アトム $s(X,Y,Z')$ の

第1引数Xおよび第2引数Yに伝搬する（$\{q_h(X), p(X,Y,Z)\} \rightarrow_{\{X,Y\}} s(X,Y,Z')$）ことを表している．

二つのsipのうち，例えば，sip1を使うものとする．マジックテンプレート法は，sip1に基づいて$S1$のルール集合$P1$を書き換えて，次のルール集合$P1^{mg}$：(5.45), ..., (5.54), を作る．

$$m\_q(a) \leftarrow . \tag{5.45}$$

$$m\_p(X) \leftarrow m\_q(X). \tag{5.46}$$

$$m\_p(X') \leftarrow m\_p(X), B(X',X). \tag{5.47}$$

$$m\_s(X,Y) \leftarrow m\_q(X), p(X,Y,Z). \tag{5.48}$$

$$m\_s(X',Y') \leftarrow m\_s(X,Y), F(X',X), G(Y',Y). \tag{5.49}$$

$$p(X,Y,Z) \leftarrow m\_p(X), A(X,Y,Z). \tag{5.50}$$

$$p(X,Y,Z) \leftarrow m\_p(X), B(X',X), C(Y',Y), D(Z',Z), p(X',Y',Z'). \tag{5.51}$$

$$s(X,Y,Z) \leftarrow m\_s(X,Y), E(X,Y,Z). \tag{5.52}$$

$$s(X,Y,Z) \leftarrow m\_s(X,Y), F(X',X), G(Y',Y), H(Z',Z), s(X',Y',Z'). \tag{5.53}$$

$$q(X,Y,Z,Z') \leftarrow m\_q(X), p(X,Y,Z), s(X,Y,Z'). \tag{5.54}$$

ここで，述語m_q, m_p, m_sはマジック述語である．

マジックテンプレート法は，緩和法と同様に，セミナイーブ法を使ってddb $S1^{mg} = (P1^{mg}, D1)$をボトムアップ評価し，生成されたga集合$IMP(S1^{mg})$の中から解を選択する．ファクト集合$D1$および定数aを先と同じデータとする．ddb $S1^{mg}$をボトムアップ評価すると，ga集合

$$\begin{aligned} IMP(S1^{mg}) = &\{m\_q(80)\} \\ &\cup \{m\_p(x) \mid 1 \leq x \leq 80\} \\ &\cup \{m\_s(x,y) \mid x = 80, 1 \leq y \leq 100\} \cup \{m\_s(x,y) \mid 1 \leq x \leq 80, 91 \leq y \leq 100\} \\ &\cup \{p(x,y,z) \mid 1 \leq x \leq 80, 1 \leq y \leq 100, 1 \leq z \leq 100\} \\ &\cup \{s(x,y,z) \mid 1 \leq x \leq 80, 91 \leq y \leq 100, 1 \leq z \leq 100\} \\ &\cup \{q(x,y,z,z') \mid x = 80, 91 \leq y \leq 100, 1 \leq z, z' \leq 100\} \end{aligned} \tag{5.55}$$

が生成される．$IMP(S1^{mg})$の中から解を選択して，式(5.43)と同じ解集合$ANS1$を得る．

緩和法のときと同様，$IMP(S1^{mg})$（式(5.55)）の中の述語pのgaの集合を$IMP_p(S1^{mg})$と記し，他の述語のgaの集合も同様に記す．$IMP(S1^{mg})$のうち$IMP_{m_q}(S1^{mg}) \cup IMP_{m_p}(S1^{mg}) \cup IMP_{m_s}(S1^{mg})$は潜在的関連集合

$$\begin{aligned} PREL1' = &\{p(x,y,z) \mid m\_p(x) \in IMP_{m\_p}(S1^{mg}), y \text{と} z \text{は任意の整数}\} \\ &\cup \{s(x,y,z) \mid m\_s(x,y) \in IMP_{m\_s}(S1^{mg}), z \text{は任意の整数}\} \\ &\cup \{q(x,y,z,z') \mid m\_q(x) \in IMP_{m\_q}(S1^{mg}), y \text{と} z \text{と} z' \text{は任意の整数}\} \end{aligned} \tag{5.56}$$

を求めるために生成したga集合であり，$IMP(S1^{mg})$のうち残りの$IMP_p(S1^{mg}) \cup IMP_s(S1^{mg}) \cup IMP_q(S1^{mg})$は，$IMP(S1)$の中のgaのうち$PREL1'$に含まれるようなgaのみを生成した結果できたga集合である（$IMP_p(S1^{mg}) \cup IMP_s(S1^{mg}) \cup IMP_q(S1^{mg}) = IMP(S1) \cap PREL1'$）．

緩和法とマジックテンプレート法の効率を比較する．どちらの方法も，主な計算時間は，与えられた問題の述語p, s, qのgaを生成するための時間である．緩和法が生成する述語sのga集合$IMP_s(S1^{MDF})$とマジックテンプレート法のそれ$IMP_s(S1^{mg})$は等しく，また，述語qのga集合$IMP_q(S1^{MDF})$と$IMP_q(S1^{mg})$も等しい．しかし，緩和法が生成する述語pのga集合$IMP_p(S1^{MDF})$に比べ，マジックテンプレート法のそれ$IMP_p(S1^{mg})$はかなり大きい（$|IMP_p(S1^{MDF})|(= 80 \times 10 \times 100) << |IMP_p(S1^{mg})|(= 80 \times 100 \times 100)$）．故に，緩和法はマジックテンプレート法より効率的になっている．

なお，マジックテンプレート法はsip1の代わりにsip2を使うこともできるが，その場合は，pの代わりにsのga集合が$|IMP_s(S1^{MDF})| << |IMP_s(S1^{mg})|$となり，この場合も，緩和法はマジックテンプレート法より効率的になる．

緩和法がマジックテンプレート法より効率的になるのは次の理由による．マジックテンプレート法において，cyclic sipに基づいてddb S^{mg}を作ると，そのボトムアップ評価ではデッドロックが生じ，gaを生成することができない．故に，マジックテンプレート法はacyclic sipのみを使う．その結果，マジックテンプレート法では，拘束が片方向にしか伝わらない．本例において，acyclicなsip1は，ルール(5.15)の本体アトム$p(X,Y,Z)$の引数Yから本体アトム$s(X,Y,Z')$の引数Yへは拘束を伝えるが，しかし，逆方向には拘束を伝えない．その結果，sのga $s(X,Y,Z)$の生成を制限するためのマジック述語$m_s(X,Y)$は$s$gaの第2引数Yについての制約を持つが，しかし，pのga $p(X,Y,Z)$の生成を制限するためのマジック述語$m_p(X)$はp gaの第2引数Yについての制約を持たない．故に，潜在的

関連集合$PREL1'$は，pの第2引数についての制約を持たない式(5.56)になる．一方，緩和法は，acyclic sipという考え方に基づかずに制約を求めるので，拘束を両方向に伝えることができ，sのga $s(X,Y,Z)$の生成を制限するための述語$rr_s(X,Y)$が第2引数Yについての制約を持つのに加え，pのga $p(X,Y,Z)$の生成を制限するための述語$rr_p(X,Y)$も第2引数Yについての制約を持つ．故に，潜在的関連集合$PREL1$は，pの第2引数についての制約を持つ式(5.44)になる．述語pについて$PREL1$は$PREL1'$よりかなり小さいため，緩和法はマジックテンプレート法より効率的になる．

本例のように，緩和法は，ある問題に対しては，マジックテンプレート法が作ることができないcyclic sipによる拘束の伝搬を反映した優れた制約（$PREL$）を作ることができ，その結果，緩和法はある問題をマジックテンプレート法よりも効率的に解くことができる．

5.3.2 例5.2

ddb $S2=(P2,D2)$のルール集合を

$$P2: \quad s(X,Y,Z) \leftarrow D(X,Y,Z). \tag{5.57}$$

$$s(X,Y,Z) \leftarrow A(X',X),B(Y',Y),C(Z',Z),s(X',Y',Z'). \tag{5.58}$$

とし，ファクト集合を$D2=D_A\cup D_B\cup D_C\cup\{D(a,a,a)\}$とする．ここで，$D_A$（$D_B,D_C$）は$A$（$B,C$）ファクトの集合である．また，問い合わせアトムを$s(X,X,X)?$とする．問題$(s(X,X,X)?,S2)$（問題2と呼ぶ）について考える．

問題2は次のように解釈できる．ファクト集合$D2$は3種類のアーク（$Aarc,Barc,Carc$と記す）をもつ有向グラフ$G=(V,D_A\cup D_B\cup D_C)$（ここで，$V=C(S2)$）を定める．$A(b,c)\in D2$のとき，節点$b$から節点$c$へのAアーク$Aarc(b,c)$が存在する．アーク$Barc$と$Carc$も同様である．$Aarc$（$Barc,Carc$）のみから成る節点$b$から節点$c$へのある路を$Apath(b,c)$（$Bpath(b,c),Cpath(b,c)$）と記し，その路の長さを$|Apath(b,c)|$（$|Bpath(b,c)|,|Cpath(b,c)|$）と記す．このとき，述語$s(X,Y,Z)$は，節点$a$から節点$X,Y,Z$へ長さの等しい3種類の路$Apath(a,X),Bpath(a,Y),Cpath(a,Z)$が存在することを示し，問い合わせアトム$s(X,X,X)?$は節点$a$から節点$X$に長さの等しい3種類の路が存在するような全ての節点Xを求めることを要求する．

緩和法の解法例を示す．

関連ddb $S2^{REL}=(P2^{REL},D2)$のルール集合$P2^{REL}$：(5.59),(5.60),(5.61),(5.62),を

$$o\_s(X,Y,Z) \leftarrow D(X,Y,Z). \tag{5.59}$$

$$o\_s(X,Y,Z) \leftarrow A(X',X),B(Y',Y),C(Z',Z),o\_s(X',Y',Z'). \tag{5.60}$$

$$r\_s(X,X,X) \leftarrow o\_s(X,X,X). \tag{5.61}$$

$$r\_s(X',Y',Z') \leftarrow r\_s(X,Y,Z),A(X',X),B(Y',Y),C(Z',Z),$$
$$o\_s(X',Y',Z'). \tag{5.62}$$

と定める．ここで，述語r_sが写像$g2(s(X,Y,Z))=r_s(X,Y,Z)$の下で関連集合$REL(s(X,X,X)?,S2)$を与える．

以下の(1)から(4)に示すように，頭部述語分割演算，述語消去演算，ルール分割演算を使って$S2^{REL}$を緩和し，写像$f2$の下で$S2^{RLXR}=(P2^{RLXR},D2)$を作る．

(1) 述語o_s,r_sの領域量を減らすために次のように頭部述語分割を行う．述語$o_s(X,Y,Z)$を$o_s_1(X,Y)\wedge o_s_2(Y,Z)\wedge o_s_3(Z,X)$に，述語$r_s(X,Y,Z)$を$r_s_1(X,Y)\wedge r_s_2(Y,Z)\wedge r_s_3(Z,X)$に分割することにより，ルール(5.59)よりルール

$$o\_s_1(X,Y) \leftarrow D(X,Y,Z). \tag{5.63}$$

$$o\_s_2(Y,Z) \leftarrow \text{ルール(5.63)の本体と同じ.} \tag{5.64}$$

$$o\_s_3(Z,X) \leftarrow \text{ルール(5.63)の本体と同じ.} \tag{5.65}$$

を作り，ルール(5.60)よりルール

$$o\_s_1(X,Y) \leftarrow A(X',X),B(Y',Y),C(Z',Z),o\_s_1(X',Y'),o\_s_2(Y',Z'),$$
$$o\_s_3(Z',X'). \tag{5.66}$$

$$o\_s_2(Y,Z) \leftarrow \text{ルール(5.66)の本体と同じ.} \tag{5.67}$$

$$o\_s_3(Z,X) \leftarrow \text{ルール(5.66)の本体と同じ.} \tag{5.68}$$

を作り，ルール(5.61)よりルール

$$r\_s_1(X,X) \leftarrow o\_s_1(X,X),o\_s_2(X,X),o\_s_3(X,X). \tag{5.69}$$

$$r\_s_2(X,X) \leftarrow \text{ルール(5.69)の本体と同じ.} \tag{5.70}$$

$$r\_s_3(X,X) \leftarrow \text{ルール(5.69)の本体と同じ.} \tag{5.71}$$

を作り，ルール(5.62)よりルール

$$r\_s_1(X',Y') \leftarrow r\_s_1(X,Y), r\_s_2(Y,Z), r\_s_3(Z,X), A(X',X), B(Y',Y), C(Z',Z),$$
$$o\_s_1(X',Y'), o\_s_2(Y',Z'), o\_s_3(Z',X'). \quad (5.72)$$
$$r\_s_2(Y',Z') \leftarrow \text{ルール}(5.72)\text{の本体と同じ}. \quad (5.73)$$
$$r\_s_3(Z',X') \leftarrow \text{ルール}(5.72)\text{の本体と同じ}. \quad (5.74)$$

を作る．

(2) 次に，ルール(5.66), (5.67), (5.68), (5.72), (5.73), (5.74)の時間量を減らすために，次のように述語消去を行う．(5.66)の本体よりZまたはZ'を含むアトムC, o_s_2, o_s_3を，(5.67)の本体よりXまたはX'を含むアトムA, o_s_1, o_s_3を，(5.68)の本体よりYまたはY'を含むアトムB, o_s_1, o_s_2を，(5.72)の本体よりZまたはZ'を含むアトム$C, o_s_2, o_s_3, r_s_2, r_s_3$を，(5.73)の本体より$X$または$X'$を含むアトム$A, o_s_1, o_s_3, r_s_1, r_s_3$を，(5.74)の本体より$Y$または$Y'$を含むアトム$B, o_s_1, o_s_2, r_s_1, r_s_2$を消去することにより，ルール(5.75), ..., (5.80)を作る．各ルールは次のようになる．

$$o\_s_1(X,Y) \leftarrow A(X',X), B(Y',Y), o\_s_1(X',Y'). \quad (5.75)$$
$$o\_s_2(Y,Z) \leftarrow B(Y',Y), C(Z',Z), o\_s_2(Y',Z'). \quad (5.76)$$
$$o\_s_3(Z,X) \leftarrow A(X',X), C(Z',Z), o\_s_3(Z',X'). \quad (5.77)$$
$$r\_s_1(X',Y') \leftarrow r\_s_1(X,Y), A(X',X), B(Y',Y), o\_s_1(X',Y'). \quad (5.78)$$
$$r\_s_2(Y',Z') \leftarrow r\_s_2(Y,Z), B(Y',Y), C(Z',Z), o\_s_2(Y',Z'). \quad (5.79)$$
$$r\_s_3(Z',X') \leftarrow r\_s_3(Z,X), A(X',X), C(Z',Z), o\_s_3(Z',X'). \quad (5.80)$$

(3) 最後に，ルール(5.75), (5.76), (5.77)の時間量を減らすために，各ルールをルール分割して，ルール(5.75)からルール

$$t_1(X,Y') \leftarrow A(X',X), o\_s_1(X',Y'). \quad (5.81)$$
$$o\_s_1(X,Y) \leftarrow t_1(X,Y'), B(Y',Y). \quad (5.82)$$

を作り，ルール(5.76)からルール

$$t_2(Y,Z') \leftarrow B(Y',Y), o\_s_2(Y',Z'). \quad (5.83)$$
$$o\_s_2(Y,Z) \leftarrow t_2(Y,Z'), C(Z',Z). \quad (5.84)$$

を作り，ルール(5.77)からルール

$$t_3(Z,X') \leftarrow C(Z',Z), o\_s_3(Z',X'). \quad (5.85)$$
$$o\_s_3(Z,X) \leftarrow t_3(Z,X'), A(X',X). \quad (5.86)$$

を作る．例えば，ルール(5.81)，(5.82)と元のルール(5.75)を比べると，$deg((5.81))(=3)$, $deg((5.82))(=3) < deg((5.75))(=4)$であるので，前者のボトムアップ評価の時間量は後者のそれより小さくなる．

以上の結果，緩和関連ddb $S2^{RLXR} = (P2^{RLXR}, D2)$はルール集合$P2^{RLXR}$：(5.63), (5.64), (5.65), (5.69), (5.70), (5.71), (5.78), (5.79), (5.80), (5.81), ..., (5.86), として，また，連言写像$f2$は$f2(o_s(X,Y,Z)) = o_s_1(X,Y) \wedge o_s_2(Y,Z) \wedge o_s_3(Z,X)$，$f2(r_s(X,Y,Z)) = r_s_1(X,Y) \wedge r_s_2(Y,Z) \wedge r_s_3(Z,X)$として得られる．

以上のようにして求めた$S2^{RLXR}$と$f2{\cdot}g2$が定める潜在的関連集合$PREL2 = \{s(X,Y,Z) \mid r_s_1(X,Y), r_s_2(Y,Z), r_s_3(Z,X) \in IMP(S2^{RLXR})\}$について考える．グラフ$G$において，節点$a$から節点$w$へ，長さの等しい2種類の路$Apath_1(a,w)$と$Bpath_1(a,w)$が存在し，かつ，長さの等しい2種類の路$Bpath_2(a,w)$と$Cpath_2(a,w)$が存在し，かつ，長さの等しい2種類の路$Cpath_3(a,w)$と$Apath_3(a,w)$が存在するような節点$w$の集合を$RelaxedANS$と記す．ここで，路$Apath_1, Bpath_2, Cpath_3$の長さは異なってもよい．このとき，$PREL2$に含まれる任意のga $s(b,c,d)$は次の四つの条件を満足する．(i) e', e'', e'''は$RelaxedANS$に属すある節点であり（$e', e'', e''' \in RelaxedANS$），(ii)節点$b$と$c$から節点$e'$へ長さの等しい路$Apath_1(b,e')$と$Bpath_1(c,e')$が存在し，(iii)節点$c$と$d$から節点$e''$へ長さの等しい路$Bpath_2(c,e'')$と$Cpath_2(d,e'')$が存在し，(iv)節点$d$と$b$から節点$e'''$へ長さの等しい路$Cpath_3(d,e''')$と$Apath_3(b,e''')$が存在する．

次に，問題2をマジックテンプレート法を使って解く場合を考える．問い合わせアトムの**修飾子**（adornment）は$s^{bbb}(X,X,X)?$ではなく$s^{bbf}(X,X,X)?$または$s^{fbb}(X,X,X)?$または$s^{bfb}(X,X,X)?$であると見なす，すなわち，全（full）sipではなく部分（partial）sipを使う方が効率的である．例えば，$s^{bbf}(X,X,X)?$と見なして潜在的関連集合$PREL2_1$を求めると，$s(X,Y,Z)$の第1引数Xと第2引数Yについての制約を表す$PREL2_1$が得られる．この$PREL2_1$に含まれるga $s(b,c,d)$は上の条件(ii)を満たすが，しかし，条件(i),(iii),(iv)は必ずしも満たさない．

更に，$s^{fbb}(X,X,X)?$と見なして潜在的関連集合$PREL2_2$を，同じく，$s^{bfb}(X,X,X)?$と

見なして潜在的関連集合$PREL2_3$を求め，$PREL2_1$と$PREL2_2$と$PREL2_3$の共通部分を新しい潜在的関連集合$PREL2'$（$= PREL2_1 \cap PREL2_2 \cap PREL2_3$）とすることもできる．このとき，$PREL2'$は条件(ii),(iii),(iv)を満たすが，しかし，条件(i)は必ずしも満たさない．

緩和法の$PREL2$とマジックテンプレート法の$PREL2_i, i = 1,2,3$，または$PREL2'$を比べると，上に述べたように，前者は後者より強い制約になっている．これは，緩和法が(1)において関連ddb $S2^{REL}$を緩和する際に頭部述語分割を使ったのに対し，一方，マジックテンプレート法はルールの書き直しの際に実質的に頭部引数消去しか使わないためである．

本例のように，緩和法は，ある問題に対しては，頭部引数消去しか使わないマジックテンプレート法では作ることができない優れた制約（$PREL$）を頭部述語分割を使うことによって作ることができ，その結果，緩和法はある問題をマジックテンプレート法よりも効率的に解くことができる．

5.3.3 例5.3

緩和法が，定数グループ化演算を使って，マジックテンプレート法では作ることができない優れた制約（$PREL$）を作る例を与える．

問題を例5.2と同じ問題2とし，関連ddbも例5.2と同じ$S2^{REL}$とする．

グラフ$G = (V, D_A \cup D_B \cup D_C)$のアーク$Aarc$（$Barc, Carc$）のみから成るグラフを$G_A = (V, D_A)$ ($G_B = (V, D_B)$，$G_C = (V, D_C)$) と記し，G_A, G_B, G_Cはサイクルを持つと仮定する．G_A（G_B, G_C）のサイクル$Acycle_i, i = 1, \ldots, l$，（$Bcycle_j, j = 1, \ldots, m, Ccycle_k, k = 1, \ldots, n$,）を計算し[Tar72]，節点$v$と$w$が$G_A$のあるサイクル$Acycle_{i'}$に属し，かつ，$G_B$のあるサイクル$Bcycle_{j'}$に属し，かつ，$G_C$のあるサイクル$Ccycle_{k'}$に属すとき（$v, w \in Acycle_{i'}, Bcycle_{j'}, Ccycle_{k'}$），$f3_c(v) = f3_c(w)$となるように定数写像$f3_c$を定める．この$f3_c$を使って$S2^{REL}$を緩和し，$S2^{RLXR'}$および写像$f3$を作る（式(5.10)）．

このようにして求めた$S2^{RLXR'}$と$f3{\cdot}g2$が定める潜在的関連集合$PREL3$について考える．$PREL3$に含まれるga $s(b,c,d)$は，ある節点eに対し3種類の路$Apath(b,e)$と$Bpath(c,e)$と$Cpath(d,e)$が存在する，という条件を満足する．但し，三つの路の長さは異なってもよい．この$PREL3$も，マジックテンプレート法は定数グループ化を使わないので，マジックテンプレート法では作ることができない．

5.4 緩和法により従来の方法を模倣する

任意の問題に対し，マジックテンプレート法が作るのと同じルール集合を緩和法が作れることを示す．

簡単のため，第5.3.1節の問題1を例に使って，問題1に対しマジックテンプレート法が作ったルール集合$P1^{mg}$：(5.45), ..., (5.54), と同じルール集合を緩和法が作れることを示す．

関連ddbのルール集合として，$P1^{REL'}$：(5.16), ..., (5.23), (5.87), (5.25), (5.88), (5.89), を使う．

(5.16)から(5.23)は第5.3.1節のそれらと同じ．

$$r\_s(X,Y,Z') \leftarrow r\_q(X,Y,Z,Z'), p(X,Y,Z), o\_s(X,Y,Z'). \tag{5.87}$$

(5.25)は第5.3.1節のそれと同じ．

$$p(X,Y,Z) \leftarrow r\_p(X,Y,Z), A(X,Y,Z). \tag{5.88}$$

$$p(X,Y,Z) \leftarrow r\_p(X,Y,Z), B(X',X), C(Y',Y), D(Z',Z), p(X',Y',Z'). \tag{5.89}$$

$P1^{REL'}$と第5.3.1節の$P1^{REL}$：(5.16), ..., (5.25), を比べた場合，$P1^{REL'}$では，r_pに限定されたpを与えるルール(5.88), (5.89)が追加され，r_sを与えるルール(5.87)の本体にはo_pではなくそのpが使われている．$P1^{REL'}$は$P1^{REL}$と比べ冗長であるが，$S1^{REL'} = (P1^{REL'}, D1)$の述語$r_p, r_s, r_q$はやはり関連集合$REL(q(a,Y,Z,Z')?, S1)$を与える．

次に，$P1^{REL'}$を以下のように緩和する．r_qの第2，第3，第4引数，r_pの第2，第3引数，r_sの第3引数を消去し，各々，述語rr_q, rr_p, rr_sとする（頭部引数消去）ことにより，ルール(5.21), (5.22), (5.23), (5.87), (5.25), (5.88), (5.89)からルール(5.21′), (5.22′), (5.23′), (5.87′), (5.25′), (5.88′), (5.89′)を作る．更に，ルール(5.21′), (5.22′), (5.23′), (5.87′), (5.25′)の本体より述語o_q, o_p, o_s, C, D, Hのアトムを消去し（述語消去），ルール(5.21″), (5.22″), (5.23″), (5.87″), (5.25″)を作る．その結果，述語o_q, o_p, o_sを頭部に持つルール(5.16), ..., (5.20)はrr_q, rr_p, rr_sのgaの生成とは無関係になるので，それらのルール自身を消去する．このようにしてできたルールの集合を$P1^{RLXR'}$：(5.21″), (5.22″), (5.23″), (5.87″), (5.25″), (5.88′), (5.89′), とする．

最後に，rr_q, rr_p, rr_sのアトムを$P1$のルール(5.11), ..., (5.15)の本体に付加することによりルール集合$P1''$を作り，変形データベース$S1^{MDF''} = (P1'' \cup P1^{RLXR''}, D1)$を得る．

ルール集合$P1'' \cup P1^{RLXR''}$は，述語名rr_q, rr_p, rr_sがm_q, m_p, m_sと異なる点を除き，

マジックテンプレート法が作ったルール集合 $P1^{mg}$: (5.45), . . . , (5.54), と等しい.

5.5 緩和法が従来の方法が解けない問題を解く例

問題はdatalog問題ではないホーン問題である（すなわち，算術述語や関数記号を含む）とする．この場合，緩和法もマジックテンプレート法も一部の問題のみを解くことができる．また，マジックテンプレート法が解ける問題であれば，第5.4節で示したように，緩和法も解くことができる．本節では，マジックテンプレート法が解くことができないあるホーン問題を緩和法が効率的に解けることを，例を挙げることにより，示す．

緩和法やマジックテンプレート法があるホーン問題を解けないのは次の理由による．緩和法やマジックテンプレート法が，ホーン問題（$(Q?, S)$ と記す）を解くために作成したddb S'（緩和法では S^{MDF}，マジックテンプレート法では S^{mg}）を，セミナイーブ法を使ってボトムアップ評価する段階を考える．S' のボトムアップ評価を完了するためには，(i) ボトムアップ評価の各ステージ（すなわち，第1章の図1.2におけるwhileループの中 $\{\Delta I = \cup_{r \in P} Eval(r, \Delta I, I) - I;\ I = I \cup \Delta I;\}$）が実行可能であること（**ボトムアップ評価の実行可能性**と呼ぶ），および，(ii) 有限回のステージで停止する（すなわち，whileループの中を有限回実行すると $\Delta I = \phi$ になる）こと（**ボトムアップ評価の停止性**と呼ぶ）が必要である．与えられた問題 $(Q?, S)$ が算術述語を含むとき，S' のボトムアップ評価で，(i) が成立しない場合や，(i) は成立するが (ii) が成立しない場合があり，また，問題 $(Q?, S)$ が関数記号を含むとき，S' のボトムアップ評価で (ii) が成立しない場合がある．故に，そのような場合，緩和法やマジックテンプレート法はホーン問題 $(Q?, S)$ を解くことができない．

本節では，マジックテンプレート法が解けないあるホーン問題を緩和法が効率的に解ける例として，マジックテンプレート法の S' が条件 (i) を満たさないにもかかわらず，緩和法の S' がそれを満たす例，および，マジックテンプレート法の S' が条件 (ii) を満たさないにもかかわらず，緩和法の S' がそれを満たす例を挙げる．

5.5.1 例5.4

ddb S の各ルールは領域制限されているが，しかし，S は算術述語を含み，かつ，問い合わせアトム $Q?$ は引数に定数を含まない問題 $(Q?, S)$ を考える．ここで，ルールの頭部に現れる全ての変数が本体にも現れるとき，そのルールは**領域制限されている**（range restricted）

と言う．

一般に，問い合わせアトム $Q?$ が引数に定数を含まない場合，マジックテンプレート法が $Q?$ に対して作るルール（seedと呼ばれる）は領域制限されない．そして，領域制限されないルールを含むddb S^{mg} をボトムアップ評価する場合，基礎アトムのみならず，非基礎アトム（すなわち，引数に変数を含むアトム）も生成する必要がある．特に，ddb S が算術述語を含む場合，ddb S^{mg} も算術述語を含み，このような場合，非基礎アトムの生成が困難なことがある（すなわち，(i) が成立しないことがある）．一方，緩和法は，このような問題 $(Q?, S)$ に対しても，全てのルールが領域制限された S^{MDF} を作れる場合があり，故に，S^{MDF} のボトムアップ評価では，たとえ算術述語を含んでいても基礎アトムのみを生成すればよい場合があり，故に，S^{MDF} のボトムアップ評価の各ステージを実行できる（すなわち，(i) が成立する）場合がある．従って，マジックテンプレート法が解くことができないあるホーン問題を緩和法が効率的に解ける場合がある．以下に例を示す．

ddb $S4 = (P4, D4)$ のルール集合を

$$P4: \quad p(X,Y) \leftarrow A(X), B(Y). \tag{5.90}$$

$$p(X,Y) \leftarrow p(X',Y'), X = X' + 10, Y = Y' + 7, 0 \le X < 20, 0 \le Y < 20. \tag{5.91}$$

とし，ファクト集合を $D4 = D_A \cup D_B$ とする．ここで，D_A（D_B）は引数が整数であるファクトの有限集合である．また，問い合わせアトムは $p(X,X)?$ とする．問題 $(p(X,X)?, S4)$（問題4と呼ぶ）について考える．

問題4において，各ルールは領域制限されているが，しかし，ルール (5.91) は算術述語 $X = X' + 10$ と $Y = Y' + 7$ と $0 \le X < 20$ と $0 \le Y < 20$ を含み，かつ，問い合わせアトム $p(X,X)?$ は引数に定数を含まないことに注意して欲しい．

問題4にマジックテンプレート法を適用すると次のルール集合 $P4^{mg}$: (5.92), . . . , (5.95), を得る．

$$m\_p(X,X) \leftarrow . \tag{5.92}$$

$$m\_p(X',Y') \leftarrow m\_p(X,Y), X' = X - 10, Y' = Y - 7, -10 \le X' < 10, \quad -7 \le Y' < 13. \tag{5.93}$$

$$p(X,Y) \leftarrow m\_p(X,Y), A(X), B(Y). \tag{5.94}$$

$$p(X,Y) \leftarrow m\_p(X,Y), p(X',Y'), X = X' + 10, Y = Y' + 7, 0 \leq X < 20,$$
$$0 \leq Y < 20. \tag{5.95}$$

ここで，m_pはマジック述語である．

初めに述べたように，問い合わせアトム$p(X,X)$?に対して作られたルール(5.92)は領域制限されていない．このルールを含むddb $S4^{mg} = (P4^{mg}, D4)$をボトムアップ評価しようとすると，例えば，ルール(5.92)と(5.93)の評価のために，以下の非基礎アトム$atom1, atom2, atom3, atom4$を生成しなければならない．

$atom1$: $m\_p(X,X)$　((5.92)より) (5.96)

$atom2$: $(m\_p(X',Y') \mid X', Y'は X' = X - 10, Y' = X - 7, -10 \leq X' < 10,$
$-7 \leq Y' < 13を満たす)$　($atom1$と(5.93)より) (5.97)
$= (m\_p(X',Y') \mid X', Y'は X' = X - 10, Y' = X - 7, 0 \leq X < 20$
を満たす) (5.98)

$atom3$: $(m\_p(X'',Y'') \mid X'', Y''は (X'' = X' - 10, Y'' = Y' - 7, -10 \leq X'' < 10,$
$-7 \leq Y'' < 13), (X' = X - 10, Y' = X - 7, 0 \leq X < 20)$を満たす)
($atom2$と(5.93)より) (5.99)
$= (m\_p(X'',Y'') \mid X'', Y''は X'' = X - 20, Y'' = X - 14, 10 \leq X < 20$
を満たす) (5.100)

$atom4$: $(m\_p(X''',Y''') \mid X''', Y'''は (X''' = X'' - 10, Y''' = Y'' - 7,$
$-10 \leq X''' < 10, -7 \leq Y''' < 13), (X'' = X - 20, Y'' = X - 14,$
$10 \leq X < 20)$を満たす)　($atom3$と$r3.4$より) (5.101)
$= (m\_p(X''',Y''') \mid X''', Y'''は X''' = X - 30, Y''' = X - 21, 20 \leq X < 20$
を満たす) (5.102)

但し，$atom4$の条件を表す式(5.102)を満足するXは存在しないので，$atom4$は捨てる．

これらのアトムの生成においては，(1) 条件を表す式(5.97)を式(5.98)に，式(5.99)を式(5.100)に，式(5.101)を式(5.102)に各々式変形し，(2) 新しく生成される$atom_j, 2 \leq j \leq 3$, がそれまでに生成された$atom_i, 1 \leq i < j$, のインスタンスであるか否かを判断するために，条件を表す式(5.96)，(5.98)，(5.100)を比較し，(3) $atom4$の条件を表す式(5.102)を満たす

Xが存在しないことを判断する，ことが必要である．

文献[Ram88]では，非基礎アトムも扱えるように拡張されたセミナイーブ法が提案され，簡単な問題に対しマジックテンプレート法の中で使われている．しかし，上のような複雑なボトムアップ評価を行うのは，このセミナイーブ法を使っても不可能と思われる（すなわち，(i)が成立しない）．

次に，問題4に対し緩和法が生成するルール集合$P4''' \cup P4^{RLXR}$: (5.103), ..., (5.110), を示す．

$$ro\_p_1(X) \leftarrow A(X). \tag{5.103}$$
$$ro\_p_2(Y) \leftarrow B(Y). \tag{5.104}$$
$$ro\_p_1(X) \leftarrow ro\_p_1(X'), X = X' + 10, 0 \leq X < 20. \tag{5.105}$$
$$ro\_p_2(Y) \leftarrow ro\_p_2(Y'), Y = Y' + 7, 0 \leq Y < 20. \tag{5.106}$$
$$rr\_p(X,X) \leftarrow ro\_p_1(X), ro\_p_2(X). \tag{5.107}$$
$$rr\_p(X',Y') \leftarrow rr\_p(X,Y), ro\_p_1(X'), ro\_p_2(Y'), X' = X - 10, Y' = Y - 7. \tag{5.108}$$
$$p(X,Y) \leftarrow rr\_p(X,Y), A(X), B(Y). \tag{5.109}$$
$$p(X,Y) \leftarrow rr\_p(X,Y), p(X',Y'), X = X' + 10, Y = Y' + 7, 0 \leq X < 20,$$
$$0 \leq Y < 20. \tag{5.110}$$

上記のルールのうち，ルール(5.103), ..., (5.108)はルール集合$P4^{RLXR}$を表し，これらのルールは，第5.3節で述べた方法により関連ddb $S4^{REL} = (P4^{REL}, D4)$を作り，ルール集合$P4^{REL}$中の述語$o_p(X,Y)$を$ro_p_1(X) \wedge ro_p_2(Y)$に頭部述語分割し，その後，領域制限を保持するようにして，不要なアトムを述語消去することにより作った．なお，$P4^{REL}$中の述語$r_p(X,Y)$は，述語名をrr_pとして，$P4^{RLXR}$中にそのまま残した．

rr_pが，マジック述語m_pが表している制約$PREL4$と同じ制約を表すことに注意して欲しい．また，$S4^{MDF}$の全てのルール(5.103), ..., (5.110)が領域制限されていること，特に，rr_pの初期値を与えるルール(5.107)が，マジックテンプレート法のm_pの初期値を与えるルール(5.92)と異なり，領域制限されていることに注意して欲しい．

セミナイーブ法による$S4^{MDF} = (P4''' \cup P4^{RLXR}, D4)$のボトムアップ評価では基礎アトムのみを生成すればよいので，ボトムアップ評価の各ステージは実行可能である（すなわ

ち，(i) が成立する）．なお，セミナイーブ法による $S4^{MDF}$ のボトムアップ評価は停止し（すなわち，(ii) が成立し），緩和法は，制約 rr_p のために，問題4に，直接，セミナイーブ法を適用した場合よりも効率的に問題を解くことができる．

このように，問題4は，マジックテンプレート法では，ボトムアップ評価の各ステージが実行できないので，解くことができないが，緩和法では効率的に解くことができる．

5.5.2 例 5.5

問題の ddb S が関数記号を含む場合を考える．但し，ddb S のボトムアップ評価は停止するものとする．

マジックテンプレート法が生成する ddb S^{mg} では，マジック述語 m_p を定めるために，元問題またはそれを簡略化した問題のトップダウン評価を模倣しようとする．このため，たとえ S のボトムアップ評価が停止しても，S^{mg} のボトムアップ評価は停止しない（すなわち，(ii) が成立しない）ことがある [Ram88, Sek89]．一方，緩和法が生成する ddb S^{MDF} では，制約を表す述語 rr_p を定めるために，通常，まず初めに元問題を簡略化した問題のボトムアップ評価を模倣し，その後，これによって得られた有限な ga 集合 IMP' の制約のもとに，元問題またはそれを簡略化した問題のトップダウン評価を模倣しようとする．このため，S^{mg} のボトムアップ評価は停止しないが，一方，S^{MDF} のボトムアップ評価は（IMP' による制約のおかげで）停止する（すなわち，(ii) が成立する）ことがある．故に，マジックテンプレート法が解くことができないあるホーン問題を緩和法が効率的に解ける場合がある．以下に例を示す．

ddb $S5 = (P5, D5)$ のルール集合を

$$P5: \quad p(X,Y) \leftarrow A(X,Y). \tag{5.111}$$

$$p(X,Y) \leftarrow p(u(X), v(Y)). \tag{5.112}$$

とする（ファクト集合 $D5$ は省略）．また，問い合わせアトムは $p(u^3(a), Y)?$ とする．ここで，u, v は関数記号，a は定数であり，$u(u(u(a)))$ を $u^3(a)$ と略記した．問題 $(p(u^3(a), Y)?, S5)$（問題5と呼ぶ）について考える．

マジックテンプレート法が生成するルール集合 $P5^{mg}$: (5.113), ..., (5.116), を示す．

$$m\_p(u^3(a)) \leftarrow . \tag{5.113}$$

$$m\_p(u(X)) \leftarrow m\_p(X). \tag{5.114}$$

$$p(X,Y) \leftarrow m\_p(X), A(X.Y). \tag{5.115}$$

$$p(X,Y) \leftarrow m\_p(X), p(u(X), v(Y)). \tag{5.116}$$

ddb $S5^{mg} = (P5^{mg}, D5)$ のボトムアップ評価は，無限個の基礎アトム $m\_p(u^3(a)), m\_p(u^4(a)), m\_p(u^5(a)), \ldots$ を生成するために，停止しない（すなわち，(ii) が成立しない）．

次に，問題5に対し緩和法が生成するルール集合 $P5' \cup P5^{RLXR}$: (5.117), ..., (5.122), を示す．

$$ro\_p(X) \leftarrow A(X,Y). \tag{5.117}$$

$$ro\_p(X) \leftarrow ro\_p(u(X)). \tag{5.118}$$

$$rr\_p(u^3(a)) \leftarrow ro\_p(u^3(a)). \tag{5.119}$$

$$rr\_p(u(X)) \leftarrow rr\_p(X), ro\_p(u(X)). \tag{5.120}$$

$$p(X,Y) \leftarrow rr\_p(X), A(X,Y). \tag{5.121}$$

$$p(X,Y) \leftarrow rr\_p(X), p(u(X), v(Y)). \tag{5.122}$$

上記のルールのうち，ルール (5.117), ..., (5.120) はルール集合 $P5^{RLXR}$ を表し，これらのルールは，第5.3節で述べた方法により関連 ddb $S5^{REL} = (P5^{REL}, D5)$ を作り，ルール集合 $P5^{REL}$ 中の述語 $o\_p(X,Y)$ および $r\_p(X,Y)$ の第2引数を頭部引数消去して作った．

ddb $S5^{MDF} = (P5' \cup P5^{RLXR}, D5)$ のボトムアップ評価は有限回で停止する（すなわち，(ii) が成立する）．また，緩和法は，制約 rr_p のために，問題5に，直接，セミナイーブ法を適用した場合よりも効率的に問題を解くことができる．

このように，問題5は，マジックテンプレート法では，ボトムアップ評価が有限回で停止しないので，解くことができないが，緩和法では効率的に解くことができる．

最後に，緩和法の適用範囲について述べる．緩和法は，第5.4節で説明したように，マジックテンプレート法が解ける任意の問題を解くことができ，かつ，本節で示したように，マジックテンプレート法が解けないあるホーン問題を解くことができるので，緩和法の適用範囲はマジックテンプレート法のそれを真に包含している．また，マジックテンプレート法の適用範囲は datalog 問題クラスと一部のホーン問題である．故に，緩和法の適用範囲は datalog 問題クラスとマジックテンプレート法よりも範囲が広い一部のホーン問題である．

5.6 例についての補足

第5.1節に述べたように，例で用いた問題（すなわち，問題1，問題2，問題4，問題5）はいずれも，説明の簡単のため，ある単純な形をしており，そのため，これらの問題は逆計数法や拡張HaNa法の簡単な変形等を使って緩和法より効率的に（但し，問題5に関しては緩和法と同程度の効率で）解くことができる．

しかし，これらの問題を少し変形すると，緩和法のマジックテンプレート法に対する優位性は保持しつつ，しかし，逆計数法や拡張HaNa法の変形等は適用できない新しい問題を作ることができる．これらの新しい問題に対しては，筆者らの知る限り，緩和法が従来のいかなる方法よりも効率的であると思われる．

以下に新しい問題を示す．

例 5.6 例5.1の問題1の変形である．ルール集合は

$$p(X,Y,Z) \leftarrow A(X,Y,Z).$$
$$p(X,Y,Z) \leftarrow B(X',X), C(Y',Y), D(Z',Z), p(X',Y',Z').$$
$$p(X,Y,Z) \leftarrow B'(X',X,W), C'(Y',Y,W), D'(Z',Z,W), p(X',Y',Z').$$
$$s(X,Y,Z) \leftarrow E(X,Y,Z).$$
$$s(X,Y,Z) \leftarrow F(X',X), G(Y',Y), H(Z',Z), s(X',Y',Z').$$
$$s(X,Y,Z) \leftarrow F'(X',X,W), G'(Y',Y,W), H'(Z',Z,W), s(X',Y',Z').$$
$$q(X,Y,Z,Z') \leftarrow p(X,Y,Z), s(X,Y,Z').$$

であり，問い合わせアトムは$q(a,Y,Z,Z')$?である．□

例 5.7 例5.2の問題2の変形である．ルール集合は

$$s(X,Y,Z) \leftarrow D(X,Y,Z).$$
$$s(X,Y,Z) \leftarrow A(X',X), B(Y',Y), C(Z',Z), s(X',Y',Z').$$
$$s(X,Y,Z) \leftarrow A'(X',X,W), B'(Y',Y,W), C'(Z',Z,W), s(X',Y',Z').$$

であり，問い合わせアトムは$s(X,X,X)$?である．□

例 5.8 例5.4の問題4の変形である．ルール集合は

$$p(X,Y) \leftarrow A(X), B(Y).$$
$$p(X,Y) \leftarrow p(X',Y'), X = X' + 10, Y = Y' + 7, 0 \leq X < 20, 0 \leq Y < 20,$$
$$10 \leq X + Y < 30.$$

であり，問い合わせアトムは$p(X,X)$?である．□

例 5.9 例5.5の問題5の変形である．ルール集合は

$$p(X,Y) \leftarrow A(X,Y).$$
$$p(X,Y) \leftarrow p(u(X), v(Y)).$$
$$p(X,Y) \leftarrow p(u(X), w(X,Y)).$$

であり，問い合わせアトムは$p(u^3(a),Y)$?である．但し，u,v,wは関数記号である．□

5.7 むすび

一般の問題を効率的に解くことを目的として，緩和法と呼ぶ新しい問い合わせ処理法を提案した．緩和法は，一般の問題を効率的に解くための従来の方法と同様に，潜在的関連集合$PREL$を求めて，それを制約として使って，解の生成に関連しない不要な基礎アトムの生成をできるだけ防ぐことで，問題を効率的に解く．しかし，緩和法は，従来の方法とは異なり，$PREL$は関連集合RELの緩和であるという考えに基づいて，初めRELを表すddbを作り，次にそのルール集合やファクト集合を簡略化して$PREL$を求める．

一般の問題を効率的に解くための従来の方法の中で最も適用範囲が広く，かつ，最も効率的であるマジックテンプレート法（ただし，アレクサンダーテンプレート法とHCT/R法もマジックテンプレート法と同じ能力をもつ）と緩和法を比較し，

1. マジックテンプレート法が解ける任意の問題に対し，緩和法が同じ$PREL$を作れる（他の$PREL$も作れる）こと，すなわち，任意の問題を緩和法がマジックテンプレート法と少なくとも同じ効率で解けること，

2. マジックテンプレート法が解けるある問題に対し，緩和法がマジックテンプレート法が作れない優れた$PREL$を作れること，すなわち，ある問題を緩和法がマジックテンプレート法より効率的に解けること，

3. マジックテンプレート法が解けないある問題を緩和法が効率的に解けること

を示した.

なお，1および3の結果として，緩和法の適用範囲がdatalog問題クラスとマジックテンプレート法よりも範囲の広い一部のホーン問題になることも説明した.

第6章

結論

本論文は，演繹データベースシステムにおける再帰的な問い合わせを効率的に処理する方法について議論した．再帰的な問い合わせ処理の効率化は演繹データベースシステムの実用化にとって重要であり，1980年代後半以降，適用範囲や効率の異なる様々な方法が提案されてきた．しかし，未だ十分ではない．本論文では，再帰的な問い合わせ処理の効率化を更に進めるために，拡張HaNa法（第2章），直積法（第3章），緩和法（第5章）と呼ぶ三つの新しい問い合わせ処理法を提案した．拡張HaNa法は適用範囲は狭いが大変効率的であり，一方，緩和法は拡張HaNa法ほどは効率的でないが適用範囲は広い．直積法は両者の中間的な適用範囲と効率をもつ．本論文では，また，直積法の内部処理に関連して，多次元直方体被覆問題と呼ぶ基本的な組合せ問題を考え，この問題を効率的に解くためのアルゴリズムも提案した．このアルゴリズムを組み込んだ直積法は，適用範囲は狭くなるが，しかし，より効率的に問い合わせを処理することができる（第4章）.

以下に，各章の結果を要約する．なお，問い合わせを処理することを問題を解くとも言う.

第2章は拡張HaNa法について述べた．再帰述語の引数の数m（≥ 2）が一般の整数である場合の同世代問題を効率的に解くための方法として，$m = 2$の場合に対して提案されたHaNa法を変形して拡張HaNa法を提案した．拡張HaNa法の最悪時間量を解析し，$m \geq 3$の場合に従来の方法の中で最悪時間量において最も効率的であるマジック集合法と比較し，$m \geq 3$の場合，通常よく生じるデータに対して，拡張HaNa法がマジック集合法より効率的であることを示した.

第3章は直積法について述べた．直積問題クラスと呼ぶdatalog問題クラスの新しい部分クラスを定義し，この問題クラスを効率的に解くための方法として直積法を提案した．直積問題クラスは拡張HaNa法の適用範囲である同世代問題クラスを真に包含している．直

積法と，直積問題クラス全体に適用できる従来の方法の中で最も効率的であるマジック集合法の効率を比較するために，直積問題の例として，同世代問題の再帰ルールを複数にした問題，および，非線形にした問題を考え，これらの問題を使って計算機実験を行った．この実験において，どちらの問題に対しても，ファクト集合がファクトを密に含む場合，直積法がマジック集合法より効率的であることを示した．

第4章は，多次元直方体被覆問題を解くアルゴリズム，および，その直積法への応用について述べた．多次元直方体被覆問題は，直積問題クラスの部分クラスとして定義した連続直積問題クラスに直積法を適用した際に，直積法の内部処理に現れる組合せ問題である．多次元直方体被覆問題は，組合せ問題の分野において有名な充足可能性問題の一般化でもある．多次元直方体被覆問題を解くために，充足可能性問題のためのDavis-Putnum法と局所探索法を多次元直方体被覆問題に素直に拡張した．また，Davis-Putnum法を直方体群の幾何学的性質を利用して拡張した，変形Davis-Putnum法と呼ぶ新しい解法も提案した．これら三つの解法の効率を調べるために計算機実験を行い，(i) 提案した変形Davis-Putnum法がある性質をもった多次元直方体被覆問題に対しDavis-Putnum法および局所探索法より効率的であること，(ii) これら三つの解法を併用すれば，様々な性質をもった多次元直方体被覆問題の多くを効率的に解けること，を示した．また，(ii)の結果として，これらの三つのアルゴリズムを直積法に組み込むことにより，連続直積問題に対しては，直積法の効率を更に高めれることを説明した．

第5章は緩和法について述べた．datalog問題クラスを包含する広い範囲の問題を効率的に解くための方法として緩和法を提案した．緩和法は，このような広い範囲の問題を効率的に解くための従来の方法と同様，制約を求め，その制約を使って問い合わせに無関係なデータの生成をできるだけ防ぐことにより，問題を効率的に解く．しかし，制約の求め方がより柔軟になっている．マジック集合法の一つであるマジックテンプレート法は，広い範囲の問題を効率的に解くための従来の方法の中で最も適用範囲が広く，datalog問題クラスと一部のホーン問題を解くことができ，かつ，最も効率的である（ただし，アレクサンダーテンプレート法とHCT/R法もマジックテンプレート法と同じ能力を持つ）．マジックテンプレート法と緩和法を比較し，マジックテンプレート法が解くことができる任意の問題を緩和法が少なくとも同じ効率で解けること，マジックテンプレート法が解けるある問題を緩和法がより効率的に解けること，マックテンプレート法が解けないある問題を緩和法が効率的に解けること，を示した．なお，緩和法の適用範囲はdatalog問題クラスと一部

のホーン問題であるが，ホーン問題部分がマジックテンプレート法より広い．

関係データベースシステムの次世代のシステムとして，演繹データベースシステムとオブジェクト指向データベースシステムが脚光を浴びて久しい．演繹データベースシステムとオブジェクト指向データベースシステムを比べた場合，残念ながら，実用化において，演繹データベースシステムは遅れをとっているように思われる．本研究が演繹データベースシステムの実用化に幾らかでも貢献できることを期待する．

文献

[AU79] Aho, A. V. and J. D. Ullman: "Universality of data retrieval languages", Proc. Sixth ACM Symp. on Principles of Programming Languages, pp.110–120(1979).

[AO89] Aly, H. and Z. M. Ozsoyoglu: "Synchronized counting method", Proc. 5th Conf. of Data Engineering, pp.366–373 (1989).

[Ban85] Bancilhon, F.: "Naive evaluation of recursively defined relations", On knowledge base management systems - Integrating database and AI systems, Blodie and Mylopoulous, Eds., Springer-Verlag, Berlin, pp.165–178 (1985).

[BMSU86] Bancilhon, F., D. Maier, Y. Sagiv and J. Ullman: "Magic sets and other strange ways to implement logic programs", Proc. 5th ACM SIGMOD-SIGACT Symp. on Principles of Database System(PODS), (1986).

[BR86] Bancilhon, F. and R. Ramakrishnan: "An amateur's introduction to recursive query processing strategies", Proc. ACM-SIGMOD Conf. on Management of Data(SIGMOD), Washington,D.C., (1986).

[BKBR87] Beeri, C., P. C. Kanellakis, F. Bancilhon, and R. Ramarkrishnan: "Bounds on the propagation of selection into logic programs", Proc. Sixth ACM Symp. on Principles of Database Systems, pp.214–226(1987).

[BR87] Beeri, C. and R. Ramakrishnan: "On the power of magic", Proc. 6th ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems (PODS), San Diego, Calif., (1987).

[BP81] Brown, C. A. and P. W. Purdom: "An average time analysis of backtracking", SIAM J. Computing, Vol.10, No.3, pp.583–593, (1981).

[DLL62] Davis, M., G. Logemann, and D. Loveland: "A machine program for theorem proving", Commun. ACM, Vol.5, pp.394–397 (1962).

[FU76] Fong, A. C. and J. D. Ullman: "Induction variables in very high-level languages," Proc. Third ACM Symp. on Principles of Programming Languages, pp.104–112 (1976).

[Fra91] Franco, J.: "Elimination of infrequent variables improves avegrage case performance of satisfiability algorithms", SIAM J. Computing, Vol.20, No.6, pp.1119–1127 (1991).

[GM78] Gallaire, H. and J. Minker: Logic and Data Bases, Plenum Press, New York (1978).

[GMN 84] Gallaire. H., J. Minker and J. M. Nicolas: "Logic and data bases: A deductive approach", ACM Comput. Surv., Vol.16, No.2, pp.153–185 (1984).

[Gel86] Gelder, A. V.: "A message passing framework for recursive query evaluation", Proc. SIG-MOD, (1986).

[GPB82] Goldberg. A., P. Purdom, and C. Brown: "Average time analysis of simplified Davis-Putnum procedures", Info. Process. Lett., Vol.15, No.2, pp.72–75 (1982).

[GSS87] Grahne. G., S. Sippu and E. Soisalon-Soininen: "Efficient evaluation for a subset of recursive queries", Proc. ACM SIGACT-SIGMOD-SIGART Symp. on PODS, pp.284–293 (1987).

[HN88] Haddad, R. W. and J. F. Naughton: "Counting method for cyclic relations", Proc. 7th ACM Symp. on PODS, pp.333–340 (1988).

[HN91] Haddad, R. W. and J. F. Naughton: "A counting algorithm for a cyclic binary query", J. Comput. & Syst. Sci., 43, 1, pp.145–169 (1991).

[HH87] Han, J. and L. J. Henschen: "Handling redundancy in the processing of recursive database queries", Proc. ACM SIGMOD Conf., pp.73–81 (1987).

[Han88] Han, J.: "Selection of processing strategies for different recursive queries", Proc. Third Intl. Conf. on Data and Knowledge Bases, Jerusalem, Israel (1988).

[Han89] Han, J.: "Multi-way counting method", Information Systems, 14, 3, pp.219–229 (1989).

[HH89] Han, J. and L. J. Henschen: "The level-cycle merging method", Proc. 1st Conf. on Deductive and Object-Oriented Databases, pp.113–129 (1989).

[HN84] Henschen, L. and S. Naqvi: "On compiling queries in recursive first-order data bases", JACM 31, pp.47–85 (1984).

[Iba89] 茨木 俊秀: アルゴリズムとデータ構造, 昭晃堂, pp.5–16 (1989).

[IW87] Ioannidis, Y. E. and E. Wong: "An algebraic approach to recursive inference", In: Kerschberg, L. (ed.) *Expert Database Systems,* Bejamin-Commings, Menlo Park, CA., pp.295–309 (1987).

[Iwa89] Iwama, K.: "CNF satisfiability test by counting and polynomial time", SIAM J. Computing, Vol.18, No.2, pp.385–391 (1989).

[Kane90] Kanellakis, P. C.: "Elements of relational database theory", In: J. V. Leeuwen editor, Handbook of theoretial computer science Vol.B — formal models and semantics —, chapter 17, pp.1073–1156, Elsevier (1990)
(鈴木 晋 訳: "関係データベース理論の構成要素", 廣瀬 健, 野崎 昭弘, 小林 孝次郎 監訳: コンピュータ基礎理論ハンドブック II — 形式モデルと意味論 —, 第17章, pp.1039–1121, 丸善 (1994)).

[Kat90] 勝野 裕文: "演繹データベースの形式的意味論", 情報処理, Vol.31, No.2, pp.198–205 (1990).

[KRS90] Kemp, D. B., K. Ramamohanarao and Z. Somogyi: "Right-, left-, and multi-linear rule transformation that maintain context information", Proc. 16th VLDB Conf., pp.380–391 (1990).

[Kiyo90] 清木 康: "演繹データベース演算処理方式", 情報処理, Vol.31, No.2, pp.215–233 (1990).

[Knu81] Knuth, D. E.: The art of computer programming: Volume 2(2nd ed.), Addison-Wesley (1981)
(中川 圭介 訳: 準数値算法/算術演算, サイエンス社, (1986)).

[Koba90] 小林 功武: "古典データベースから演繹データベースへ", 情報処理, Vol.31, No.2, pp.189–197 (1990).

[Loz85] Lozinskii, E.: "Evaluating queries in deductive databases by generating", Proc. 11th Int. Joint Conf. on Artificial Intelligence, (1985).

[MS81] Mckay, D. and S. Shapiro: "Using active connection graphs for reasoning with recursive rules", Proc. 7th Int. Joint Conf. on Artificial Intelligence, (1981).

[Min88] Minker, J.: "Persectives in deductive databases", J. Logic Program, Vol.5, No.1, pp.33–60 (1988).

[MYHI89] Miyazaki, M., K. Yokota, H. Haniuda and H. Itoh: "Horn clause transformation by restrictor in deductive databases", J. Inf. Process(IPSJ), Vol.12, No.3, pp.266–279 (1989).

[MI93] 宮崎修, 岩間一雄: "CNF 論理式に対する局所探索法の評価", 情処学アルゴリズム研報, 32-13, pp.97–104 (1993).

[MS90] 宮崎 収兄, 世木 博久: "演繹データベースの問合せ処理", 情報処理, Vol.31, No.2, pp.216–224 (1990).

[Mori95] 森下 真一: "演繹データベースの問合せ最適化技術", 情報処理, Vol.36, No.6, pp.545–552 (1995).

[MP91] Mumick, I. S. and H. Pirahesh: "Overbound and right-linear queries", Proc. 9th ACM Symp. on Principles of Database Systems(PODS), pp.127–141 (1991).

[Nau88] Naughton, J. F.: "Compiling separable recursions", ACM SIGMOD Intl. Conf. on Management of Data, pp.312–319(1988).

[NRSU89a] Naughton, J. F., R. Ramakrishnan, Y. Sagiv and J. D. Ullman: "Argument reduction by factoring", Proc. VLDB, pp.173–182 (1989).

[NRSU89b] Naughton, J. F., R. Ramakrishnan, Y. Sagiv and J. D. Ullman: "Efficient evaluation of right-, left-, and multi-linear rules", Proc. ACM-SIGMOD Conf. on Management of Data, pp.235–242 (1989).

[NK88] 西尾 章治郎, 楠見 雄規: "演繹データベースにおける再帰的な問合せ評価法", 情報処理, Vol.29, No.3, pp.240–255 (1988).

[OYO93] 大柳俊夫, 山本雅人, 大内東: "陰的列挙法に基づくSATアリゴリズム", 情処学論, Vol34, No.12, pp.2464–2473 (1993).

[PS77] Paige, R. and J. T. Schwartz: "Reduction in strength of high level operations," Proc Fourth ACM Symp. on Principles of Programming Languages, pp.58–71 (1977).

[PB85] Purdom, P. W. and C. A. Brown: "The pure literal rule and polynomial average time", SIAM J. Computing, Vol.14, No.4, pp.943–953 (1985).

[Pur87] Purdom, P. W.: "Polynomial-average time satisfiability problems", Information sciences, 41, pp.23–42 (1987).

[Pur90] Purdom, P. W.: "A survey of average time analyses of satisfiability algorithms", J. Information processing, Vol.13, No.4, pp.449–455 (1990).

[Ram88] Ramakrishnan, R.: "Magic templates: A spellbinding approach to logic programs", Proc. 5th Int. Conf. and Symp. on Logic Programming(ICLP/SLP), Seattle, Wash., (1988).

[RU95] Ramakrishnan, R. and J. D. Ullman: "A survey of deductive database systems", J. Log. Program, Vol. 23, No. 2. pp.125–149 (1995).

[RLK86] Rohmer, J., R. Lescoeur and J. M. Kerisit: "The Alexander method, a technique for the processing of recursive axioms in deductive database", New Generation Computing, 4(3), (1986).

[SZ86] Sacca, D. and C. Zaniolo: "On the implementation of a simple class of logic queries for databases", Proc. 5th ACM SIGACT-SIGMOD Symp. on PODS, pp.16–23 (1986).

[SZ87] Sacca, D. and C. Zaniolo: "Magic counting methods", Proc. ACM SIGMOD Intern. Conf. on Management of Data, pp.49–59 (1987).

[Sek89] Seki, H.: "On the power of Alexander templates", Proc. 8th ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems(PODS), Philadelphia, Penn., (1989).

[SLM92] Selman, B., H. Levesque, and D. Mitchell: "A new method for solving hard satisfiability problems", Proc. AAAI-92, pp.440–446 (1992).

[SPS87] Spaccamela, A. M., A. Pelaggi and D. Sacca: "Worst-case complexity analysis of methods for logic query implementation", Proc. 6th ACM Symp. on Principles of Database Systems, pp.294–301 (1987).

[SIK90] Suzuki, S., T. Ibaraki and M. Kishi: "Reverse counting method for linear recursive query with many cyclic extensional predicates", Proc. IASTED international symposium Artificial Intelligence Applications and Neural Networks (AINN'90), pp.22–25 (1990).

[SIK91] Suzuki, S., T. Ibaraki and M. Kishi: "Using relaxation techniques to evaluate queries in deductive databases", Proc. 2nd Int. Conf. on Database and Expert Systems Applications (DEXA'91), pp.67–72 (1991).

[SIK92] 鈴木 晋, 茨木 俊秀, 岸 政七: "多変数同世代問題に対する問合せ評価法", 電子情報通信学会論文誌, Vol.J75-D-I, No.10, pp.934–943 (1992).

[SIK93] Suzuki, S., T. Ibaraki and M. Kishi: "Query evaluation of the same generation problem with many variables（SIK 92の英訳）", Systems and computers in Japan, Vol.24, No.10, pp.1–14 (1993).

[SIK94] 鈴木 晋, 茨木 俊秀, 岸 政七: “緩和法による演繹データベースの問合せ評価”, 情報処理学会論文誌, Vol.35, No.11, pp.2437–2455 (1994).

[SI97] 鈴木 晋, 茨木 俊秀: “多次元直方体被覆問題および充足可能性問題を解くアルゴリズム”, 電子情報通信学会論文誌, Vol.J80-D-I, No.7, pp.591–604 (1997).

[Tar72] Tarjan, E.: “Depth first search and linear graph algorithms”, SIAM J.Compt., Vol.1, No.2, pp.146–160 (1972).

[Ull85] Ullman, J. D.: “Implementation of logical query languages for databases”, TOLD, 10(3), pp.289–321 (1985).

[Ull88] Ullman, J. D.: Principles of database and knowledge-base systems, Computer Science Press, Vol I (1988).

[Ull89] Ullman, J. D.: Principles of database and knowledge-base systems, Computer Science Press, Vol II (1989).

[Ull90] Ullman, J. D.: “The theory of deductive database systems”, Dig. Pap. COMPCON, Vol.1990, No. Spring, pp.496–502 (1990).

[UZ90] Ullman, J. D. and C. Zaniolo: “Deductive databases: Achievements and future directions”, ACM SIGMOD Rec., Vol.19, No.4, pp.75–82 (1990).

[Ull91] Ullman, J. D.: “A comparison between deductive and object-oriented database systems”, Lect. Notes Comput. Sci., Vol.566, pp.263–277 (1991).

[Vie86] Vieille, L.: “Recursive axioms in Deductive Databases. The Query/Subquery Approach”, Proc. First Int. Conf. on Expert Database Systems, Charleston, (1986).

[Zani90] Zaniolo, C.: “Deductive database — Theory meets practice”, Lect. Notes Comput. Sci., Vol.416, pp.1–15 (1990).