社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2004－AL－94 (14)
2004／3／19

混合探索による順序を用いた Jones 多項式の計算手法

内海 友雄 *　　今井 桂子 †
* 中央大学大学院 理工学研究科 情報工学専攻
† 中央大学 理工学部 情報工学科

概要： 結び目の不変量の 1 つに Jones 多項式があるが, これを計算することは $\#P$-困難であることが知られている. 二分決定グラフ (Binary Decision Diagram; BDD) を用いた Jones 多項式の計算においては, BDD の幅が Jones 多項式計算の計算量を決定する. 本稿では BDD の幅が結び目の Tait グラフの混合探索数と関係があることを示す. 混合探索数の近似解を計算することのできる外平面グラフを Tait グラフとして持つ結び目に対して, Tait グラフの枝数を m とした時, BDD を用いることによって Jones 多項式が $O(m^2)$ で計算できることを示す.

Computation of the Jones Polynomials using mixed search

Tomoo UTSUMI *　　Keiko IMAI †
* Information and System Engineering Course,
Graduate School of Science and Engineering, Chuo University
† Department of Information System and Engineering, Chuo University

Abstract : The Jones polynomial is one of the invariants in knot theory, and it is known that computing the Jones polynomial of a knot is $\#P$-hard. We consider the problem of computing the Jones polynomials and we use Binary Decision Diagram(BDD) for this problem. We present that the width of the BDD has a strong connection with the mixed search number of the Tait graph for the knot. We show the Jones polynomial for a knot whose Tait graph is outerplanar can be computed in $O(m^2)$ time, where m is the number of edges in the Tait graph. Computational results are also shown.

キーワード: knot, Jones polynomial, Binary Decision Diagram, mixed search number.

1 はじめに

結び目・絡み目といった位相に関する構造を, 計算量の観点から研究する分野として, 計算位相幾何学と呼ばれる分野が形成されてきている.

Jones 多項式は結び目の不変量であり, 自明な結び目と同じ多項式をもつ自明でない結び目はまだ見つかっていないことから, 結び目の判別の能力が一般に優れていると言われている. 2 つの結び目の同値性判定については有効なアルゴリズムが知られておらず, 2 つの結び目の Jones 多項式を計算して, それらが違う場合には同値でないという計算による証明法が重要となる. しかし, D. J. A. Welsh により Jones 多項式の計算量は $\#P$-困難であることが示されており [8], 最悪指数時間かかると思われている.

L. H. Kauffman により結び目のリンク・ダイアグラム D から Jones 多項式を計算する方法が考案された. ここで, リンク・ダイアグラムの交点数を $c(D)$ として, 一般の結び目に対して Jones 多項式を計算するアルゴリズムとしては, 関根らによる $O(2^{O(\sqrt{c(D)})})$ 時間アルゴリズムが知られている [5].

しかし, 入力するリンク・ダイアグラムに制限を加えると, 多項式時間で計算できる場合がある. 1999 年に Mighton により, Tait グラフの tree width が高々 2 であるとき Kauffman bracket 多項式を $O(c(D)^4)$ 時間で計算できることが示された [4]. また, 2000 年に Makowsky は, tree width が定数のとき colored Tutte 多項式を計算する多項式時間アルゴリズムを示し [3], それを利用することにより, Tait グラフの tree width が定数のとき $c(D)$ の多項式時間で計算できることを示した. また, 原・谷・山本が Arborescent link を $O(c(D)^3)$ 時間計算できることを示した [2].

本稿では, [5] のアルゴリズムの計算順に, 混合探索を行なう際に用いる枝探索順を利用し, 混合探索数が定数となるようなグラフに対しては Jones 多項式が $O(c(D)^2)$ で計算できることを示す.

2 結び目と絡み目

結び目とは, 円 S (図 1) を 3 次元 Euclid 空間 $\mathbb{R}^3$ へ埋めこんだものである. 絡み目 L とは, 結び目の有限個の集まりのことをいう. 3 次元の絡み目 L を 2 次元平面に射影したものを考える. 平面上に得られた絡み目の像において多重点は 2 重

点だけになるように射影する．これをリンク・ダイアグラム D という (図 2).

図 1. 結び目

図 2. リンク・ダイアグラム

2 つの紐の絡み目が位相的に同じであるかどうかを判定するには相異判定が容易な代数系を見つけて，それらを比較して，判定することが必要となる．これらの数量や代数系を結び目の不変量という．

3 Jones 多項式の計算

結び目の不変量である Jones 多項式について述べる．まず，Jones 多項式と関係の深い Kauffman bracket 多項式 (以下 Bracket 多項式) について説明する．

3.1 Bracket 多項式

Bracket 多項式 $< D >$ は与えられた絡み目のリンク・ダイアグラム D に次の 3 つの規則を適用して得られる 1 変数 A の多項式である．

(i) $< U >= 1$

(ii) $< DU >= -(A^2 + A^{-2}) < D >$

(iii) $< \asymp >= A < \smile\!\frown > + A^{-1} <)(>$

$< \asymp >= A <)(> + A^{-1} < \smile\!\frown >$

ただし，U は自明な結び目，DU は自明な結び目との結合を表すとする．また (iii) はリンク・ダイアグラムの各交点で局所的に適用される．

3.2 Jones 多項式と Bracket 多項式の関係

絡み目のリンク・ダイアグラムにおいて，各結び目で同じ向きになるようにリンク・ダイアグラムに向きをつける (図 3).

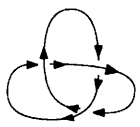


図 3. 向きをつけた絡み目

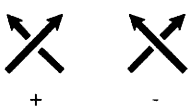


図 4. 交差の向き

向きをつけたリンク・ダイアグラムの交差に対して，図 4 に従って交差に符号をつける．向きのついた絡み目のねじれの数 $w(D)$ とは，向きのついたリンク・ダイアグラム D における交点の符号 ($+1$ もしくは -1) の総和で表される．任意の向きのついた絡み目 L のリンク・ダイアグラム D に対して，Jones 多項式 V_L と Bracket 多項式 $< D >$ は次式のような関係にある．

$$V_L(A^{-4}) = (-A^3)^{-w(D)} < D >$$

以下では，Jones 多項式を求める代わりに Bracket 多項式を求める方法について議論する．

3.3 Bracket 多項式の再帰計算式

隣り合うリンク・ダイアグラムの領域が同じ色にならないように白黒の 2 色によって色分けすることができる．このとき，色分けされた領域のうち黒の部分を点 v とする．もし 2 つの黒の領域が交点を境界で共有しているならば，黒の領域を表す点を線分で結びそれを枝 e とし，向きのついた絡み目 L における交点の符号を枝 e につける．このようにして得られるグラフを Tait グラフ S (以下，グラフは Tait グラフを意味する) とよぶ．グラフ S から枝 e を削除して得られるグラフを $S\backslash e$, S から枝 e を縮約して得られるグラフを S/e で表す．両端が同一頂点であるような枝をループ，その枝を削除するとグラフの連結成分数が 1 増えるような枝をコループとよぶことにする．

グラフ S に対して枝の縮約・削除公式が成り立つことが Thislethwaite [7] によって示された．それを用いると次の再帰計算式が得られる [5]. この再帰計算式を用いて，計算グラフを展開した例を図 5 に示す．

$$< S >= \begin{cases} -A^{-3} < S/e > & e \text{ が正のコループ} \\ -A^{3} < S/e > & e \text{ が負のコループ} \\ -A^{3} < S\backslash e > & e \text{ が正のループ} \\ -A^{-3} < S\backslash e > & e \text{ が負のループ} \\ A < S/e > + A^{-1} < S\backslash e > & \text{それ以外で } e \text{ が正} \\ A < S\backslash e > + A^{-1} < S/e > & \text{それ以外で } e \text{ が負} \end{cases}$$

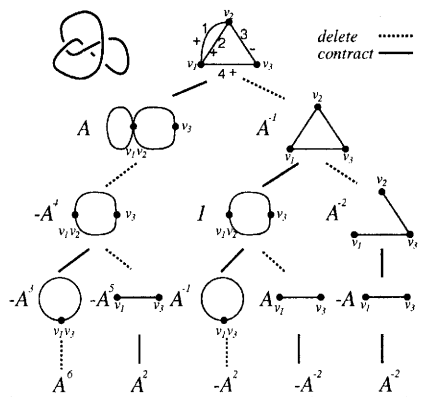


図 5. 再帰計算を用いた Bracket 多項式の計算例

3.4 二分決定グラフ

グラフ S の枝集合 E の枝順序を $e_1, e_2, ..., e_m$ ($|E| = m$) とし, レベル i において枝 e_i を縮約または削除する. このときレベル i において既に縮約・削除を行った枝集合を $E_i = \{e_1, e_2, ..., e_i\}$, その補集合を $\overline{E_i} = \{e_{i+1}, e_{i+2}, ..., e_m\}$ とする. レベル i のエリミネーション・フロントとは, E_i に属する枝および $\overline{E_i}$ に属する枝の両方に接続する頂点集合のことである. レベル i のエリミネーション分割とはレベル i のエリミネーション・フロントに属する頂点において, 縮約により統一された頂点を同値類とみなした頂点分割のことである. このとき, 次の定理が成り立つ. ここでマイナーとは縮約・削除を繰り返して得られるグラフのことである.

定理 1 H_1, H_2 を枝集合 $\overline{E_i}$ からなる S のマイナーとする. H_1 と H_2 が枝の順序も含めて同型である必要十分条件は, H_1 と H_2 のエリミネーション分割が等しいことである.

符号付きグラフ S においては, 枝 e の縮約・削除をおこなっても残りのどの枝の符号も変化しない. 従って枝の順序も含めて同型なマイナーは, 各枝の符号も含めて同型である. 上記の定理 1 をもとに, 枝の順序も含めて同型なマイナーをすべて統一して Bracket 多項式を計算することができる (図 6). このように枝の順序を含めて同型なマイナーを統一した計算過程のグラフは, 二分決定グラフ (Binary Decision Diagram) と関係がある. そこで, この計算過程のグラフを単に BDD とよぶ. また, 各レベルのマイナーの数はエリミネーション・フロントの数のベル数以下になる.

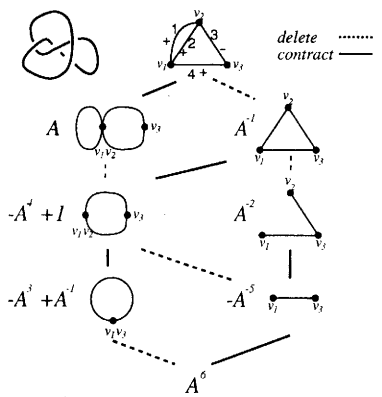


図 6. BDD を使った計算例

4 アルゴリズム

BDD を用いた Bracket 多項式の計算は図 7 の手順でおこなう. 最初に, 符号付きグラフ S の枝に順序を付けておく. $\widetilde{G}$ は操作しているマイナー, e_i は操作している枝, また, u, u' は e_i の端点とする. f_i はエリミネーション・フロントの数とする. $sign_i$ は, e_i の枝の符号を $\{1, -1\}$ で表現したものである.

この手順で計算したときの計算時間を考える. ただし, ここではグラフが連結グラフとなることより $|V| \leq 2m = |E|$ を用いている ($|V| \geq 2$).

エリミネーション・フロントを見つけるのは, これまでに操作した枝と操作していない枝の両方の枝が持つ頂点を見つけるだけなので $O(m)$ である. ループの判定は, $\widetilde{G}$ のエリミネーション分割で, u, u' が同じ集合に入っているかを判定すればよいので, $O(f)$ である. また, コループの判定は, e_i を削除したときに連結成分数が 1 増えるかの判定であるので, $\widetilde{G}\backslash e_i$ において u から u' へのパスが存在するかを判定すればよい. 従って, $O(m)$ 時間である.

各マイナー $\widetilde{G}$ における多項式は最高次数 $3m$, 最低次数 $-3m$ であり, 最大で $O(m)$ 個の項の多項式を持つことがわかる. よって, その和や単項式との積は $O(m)$ である. また, 2 同型なマイナーの探索は, そのレベルでの B_{f_i} 個のマイナーとエリミネーション分割が等しいかどうかを調べるので, $O(f_i B_{f_i})$ となる. ここで, $ef(G) = \max f_i (1 \leq i \leq m)$ とすると, このアルゴリズム総計算量は, $O(m^2 B_{ef(G)} + ef(G) m B^2_{ef(G)})$ 時間となる. 一般にベル数 B_n は $o(n^n)$ であるが, エリミネーション・フロントの数 $ef(G)$ が定数となるような絡み目に対しては $B_{ef(G)}$ が定数となり, Bracket 多項式を $O(m^2)$ 時間で計算できる.

5 平面分離定理

BDD を用いて Jones 多項式を計算する場合, 異なる枝順序を用いると BDD の大きさが変わる. 従って, エリミネーション・フロントの数をなるべく少なくするような計算順が重要となる. 関根ら [5] によると連結な平面グラフ G に対して, 平面分離定理を用いて枝順を決めると, 各レベルで $O(\sqrt{n})$ 個のエリミネーション・フロントで計算することができる.

平面分離とはすべて頂点を頂点集合 A, B, C に次の 2 つの条件を満たすように分離するものである.

```
Input: 順序の付いた グラフ S
Output: < D >
G̃ := S
S_0 := {G̃}
T(S_0) := 1
begin
  for i := 1 to n do
  begin
    エリミネーション・フロントを見つける
    S_i := 0;
    for S_{i-1} のそれぞれのマイナーを G̃ とする do
    begin
      if G̃ 中で e_i がループ then
        M(G̃) := {G̃\e_i}
        T(G̃\e_i) := -A^{3×sign_i} × T(G̃)
      else if G̃ 中で e_i がコループ then
        M(G̃) := {G̃/e_i}
        T(G̃/e_i) := -A^{-3×sign_i} × T(G̃)
      else M(G̃) := {G̃/e_i, G̃\e_i}
        T(G̃/e_i) := A^{1×sign_i} × T(G̃)
        T(G̃\e_i) := A^{-1×sign_i} × T(G̃)
      for M(G̃) のそれぞれのマイナー G_{e_i} do
      begin
        if S_i 中に G_{e_i} と 2 同型なマイナー Ĝ が
        存在する then
          T(Ĝ) := T(Ĝ) + T(G̃_{e_i})
        else
          S_i に G_{e_i} を追加する
      end
    end
  end
end
```

図 7. 計算手順

- A に属する頂点と B に属する頂点は辺で結ばれない.
- $|A| < \frac{2}{3}$, $|B| < \frac{2}{3}$, $|C| < 2\sqrt{2n}$.

3 つに分離された頂点集合うち A と B に属するものに対して, 再帰的に平面分離を用い, 頂点順を $A < B < C$ とする. そして, 枝に対して辞書式順の昇順に順序を決める. これが平面分離を用いた枝順である.

6 混合探索数

混合探索とは, グラフの探索ゲームの探索法の 1 つである. 探索ゲームとは, なるべく少ない石数ですべての枝を探索する問題である. 探索した枝と未探索な枝が石の置いていない頂点で接すると探索した枝が未探索な枝に戻ることに注意する. 既探索な枝が未探索な枝に戻らないような石の置き方を一方向走査戦略とよぶ. この一方向走査戦略による石の置き方が最適となる石の置き方を必ず含んでいることが, [9] で証明されている. 混合探索数はグラフ G を探索するのに必要な最小の石の数のことである. グラフ G の混合探索数を $ms(G)$ で表す. また, 混合探索数を一般のグラフに対して求めるのは NP-困難なことも知られている.

混合探索では次の 3 つの方法で, 枝を探索する.

M1:	ある枝 $e = (u, v)$ の u, v に石を置く.
M2:	石を 2 つの u に移動する. つぎに 1 つの石を e に沿って v に移動する.
M3:	u に石があり, u に接続する e 以外のすべての枝が既探索であるとする. u の石を e に沿って v に移動する.

6.1 グラフに対する操作

グラフ G に対して混合探索数を求める際に行なうことのできる操作として, 次にあげられる 4 つの操作がある. 1 と 2 の操作を行なったとき, 混合探索数は変化しない. 3 と 4 の操作を行なったとき, 変形したグラフの混合探索数を求め, それぞれの操作をもとに戻し, 必要な石数を数え直せばグラフ G の混合探索数を求めることができる.

1. グラフ G が自己閉路 $e(v, v)$ をもつとき, v に石を置いたときに自己閉路 e を探索することにし, e をグラフ G 上から削除する.
2. グラフ G が並列枝 $e_i(u, v)$, $e_j(u, v)$ をもつとき, 枝 $e_i(u, v)$ を探索したときに, $e_j(u, v)$ も探索することにし, 枝 e_j をグラフ G 上から削除する.
3. グラフ G が次数 1 の頂点 v をもつとき, その頂点に接続する枝を $e(u, v)$ とする. $e(u, v)$ は u に石を置いた次の操作で探索することにし, $e(u, v)$ を削除する. $e(u, v)$ を削除したグラフを G_0 とすると $ms(G) - 1 \leq ms(G_0) \leq ms(G)$ である.
4. グラフ G が次数 2 の頂点 v をもつとき, その頂点 v を削除し, 両隣の頂点を枝で結ぶ. 例えば, 頂点 v がもつ枝を $e_i(u, v)$, $e_j(v, w)$ とし, それらのグラフを $e_k(u, w)$ に置き変える. 置き換えたグラフをグラフ G_0 とする. G_0 で混合探索をし, e_k を探索するとき, G の e_i, e_j を探索することにする. このとき, $ms(G) - 1 \leq ms(G_0) \leq ms(G)$ である.

6.2 混合探索数とエリミネーション・フロント

本稿で扱うエリミネーション・フロントと混合探索の関係を説明する. 混合探索での枝の探索をエリミネーション・フロントでの枝の操作と考える. 混合探索では枝を探索する際に, 石を置くか, 石を移動させなくてはならない. しかし, エリミ

ネーション・フロントでは枝の操作であるので, 石を移動したあとの状態だけを考えればよい. そこで, 一方向走査戦略を用いた混合探索に次の探索を足したものがエリミネーション・フロントの最大数 $ef(G)$ を求める探索法といえる.

> M4: u に石があり, 枝 $e(u,v)$ を探索するとき, 頂点 v が次数 1 の頂点のときは, 石の移動なしにその枝は既探索とする.

この操作によって $ms(G)$ と $ef(G)$ には, $ef(G)-1 \leq ms(G) \leq ef(G)$ の関係が成り立つことがわかる.

一般のグラフに対して混合探索数を計算することはできない. そこで, 次に混合探索数を見積もることのできる, 木や外平面グラフについて議論する.

7 Tait グラフが木となる場合

Tait グラフが木になるような結び目の Jones 多項式を求める.

7.1 木の混合探索数

高橋ら [6] によって, 木に対する混合探索数を求めるアルゴリズムが提案されている.

木の混合探索数について, 次の定理が成り立つ.

定理 2 木を T とし, $ms(T) = k+1$ が成り立つとする. ある頂点 u について, $T \backslash u$ は混合探索数 k となる部分木を 3 つ以上もつ頂点 u が存在する.

このことから, 木を親から部分木に葉となるまで分解し, 葉から親まで部分木ごとの混合探索数を求めることによって混合探索数を求めることができる [6].

7.2 木となる場合の Jones 多項式

木に対して Jones 多項式を計算することを考えたとき, すべての枝がコループとなることがわかる. このことから, 木に対しては, 混合探索数に関わらず, 前述アルゴリズムで $O(m^2)$ で計算することができる. しかし, すべてがコループであるので, $+$ と $-$ の数だけ数えて, それぞれを $|+|$, $|-|$ とし, $-A^{-3\times|+|} \times -A^{3\times|-|}$ を計算すれば十分である. このことから $O(m)$ の計算時間で計算することができる.

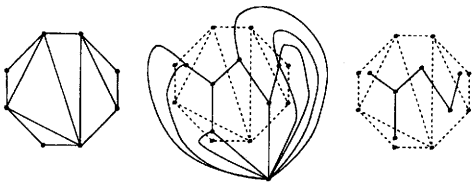
8 Tait グラフが外平面グラフとなる場合

木の混合探索数が求められることを利用して, Tait グラフが外平面グラフになるような結び目の Jones 多項式を求める.

8.1 外平面グラフの混合探索数

外平面グラフ G とは, 平面に枝の交差なしに描画されたグラフにおいて, すべての頂点が辺に囲まれていないようなグラフ (図 8 (a)) のことである.

dual グラフ (図 8 (b)) とは, 2 つのグラフ $G=(V,E)$, $G^*=(V^*,E^*)$ の枝集合 E, E^* の間に 1 対 1 対応が存在し, その対応のもとで G のタイセットと G^* のカットセットとが 1 対 1 対応するグラフである. weak dual グラフ (図 8 (c)) とは, 平面グラフ G の dual グラフ G^* において, 描画上 G の外部の領域に対応する G^* の頂点とそれに付属する枝を取り除いたものである.



(a) (b) (c)

図 8. 外平面グラフ (a) と dual グラフ (b), weak dual グラフ (c)

H. L. Bodlaender ら [1] によって, 二連結な外平面グラフに対して, パス幅の近似解を求めるアルゴリズムが提案さた.

パス幅 pw とは次の 4 つの条件を満たす頂点集合 $X_1,\cdots,X_r$ にグラフを分解し, $pw=\max\{|X_i|-1, 1\leq i\leq r\}$ となる値である.

> **1:** $X_i \not\subseteq X_j (i \neq j)$.
> **2:** $\cup_{1\leq i\leq r} X_i = V(G)$.
> **3:** 任意の辺 $(u,v)\in E(G)$ に対して, $u,v\in X_i$ であるような i が存在する.
> **4:** $X_l \cap X_n \subseteq X_m \quad (1\leq l\leq m\leq n\leq r)$.

外平面グラフのパス幅を求めるアルゴリズム (図 9) は二連結な外平面グラフの weak dual グラフが木になることを用いて, 外平面グラフを G^*, その双対グラフを G としたとき, $pw(G) \leq pw(G^*) \leq 2pw(G)+2$ となる $pw(G^*)$ を求める. 高橋ら [6] によると, パス幅と混合探索数には $pw(G) \leq ms(G) \leq pw(G)+1$ の関係が成り立つので, $ms(G) \leq ms(G^*) \leq 2ms(G)+4$ となる.

よって, G^* の混合探索数は $ms(G)$ の 2 倍程度で押えられる.

Step 1: グラフ G^{**} の次数 2 の頂点, 自己閉路と並列辺となる辺を削除し, それを G^* とする. G^* は外平面グラフとなる.
Step 2: グラフ G^* の dual グラフ G を求める.
Step 3: グラフ G のパス幅を求める.
Step 4: グラフ G のパス幅に基づいて, グラフ G^* のパス幅の近似解を求める.
Step 5: グラフ G^* に Step 1 で削除し辺, 自己閉路と並列枝を戻す.

図 9. 外平面グラフのパス幅

8.2 外平面グラフの場合の Jones 多項式

8.1 節より, パス幅に対する近似アルゴリズムに従って探索する枝順序を決めると, その時の $ef(G)$ は $pw(G)$ の 2 倍程度になることがわかる. そこでこの順序を枝順として用いることにする.

外平面グラフでは, 木のときと違いすべてがコループとなるわけではない. このことから, 外平面グラフに対して, この枝順を用いることには意味があるといえる.

9 計算機実験

Tait グラフが外平面グラフとなるような結び目の Jones 多項式を計算する実験を行なった. 8.2 節で述べたように枝順を決め, 二分決定グラフを用いた Jones 多項式計算を実装した. 外平面グラフ G^* の $ef(G^*)$ が一定となるグラフに対して, 計算時間と枝数の関係を調べた. また, 計算時間に混合探索を求める時間は入っていない. データ構造を工夫すれば, 頂点数を n として, 木と外平面グラフの混合探索数は $O(n \log n)$ で求めることができる.

図 10 は横軸にグラフの枝数 (m), 縦軸に計算時間 $(time)$ をとった両対数グラフである. 実線は混合探索を用いた枝順による実験結果の近似直線, 点線は平面分離定理を用いた枝順による実験結果の近似直線である. 実線は $time = m^{1.97} + b$, 点線は $time = m^{3.51} + b'$ である (b, b' は定数). このことから混合探索を用いた枝順では外平面グラフの $ef(G)$ が一定なとき, Jones 多項式を求めるのに $O(m^2)$ の計算量であることが確認できる. また, 平面分離定理を用いた枝順より小さい計算量で計算できていることも確認できた.

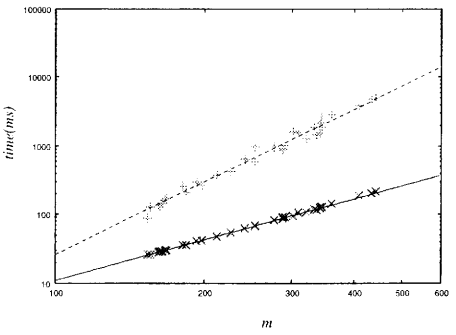


図 10. 計算機実験の結果

10 結論

本稿で扱ってきた枝数 m はリンク・ダイアグラム D の交差数 $c(D)$ と等しい. Tait グラフの混合探索を求めるのが $O(c(D)^2)$ 時間以下であり, Tait グラフの混合探索数が一定となる結び目の Jones 多項式の計算量は $O(c(D)^2)$ である.

参考文献

[1] H. L. Bodlaender, F. V. Fomin: Approximation of pathwidth of outerplanar graphs, *Proceedings of the 27th International Workshop on Graph-Theoretic Concepts in Computer Science*, vol.2204 of LNCS, pp.166-176, 2001.

[2] 原正雄, 谷聖一, 山本慎: Arborescent link の Jones 多項式の計算アルゴリズムについて, 2001 年度冬の *LA* シンポジウム, 2002.

[3] J. A. Makowsky: Colored Tutte polynomials and Kauffam brackets for graphs of bounded tree width, *submitted to Combinatorics*, Probability and Computation, 2000.

[4] J. Mighton: Knot Theory on Biparitite Graphs. PhD thesis, Department of Mathmetics, University of Toronto, Toronto, Canada, 1999.

[5] 関根京子, 今井浩, 今井桂子: Jones 多項式の計算, 日本応用数理学会論文誌, Vol.8,No.3, pp.341-354, 1998.

[6] Atsushi Takahasi, Shuichi Ueno, Yoji Kajitani: Mixed search and proper-path-width, *Theoretical Computer Science*, 137, pp.253-268, 1995.

[7] M. B. Thisletwaite: A Spanning Tree Expansion of the Jones Polynominal, *Topology* Vol.26, pp.297-309, 1987.

[8] D. J. A. Welsh: Complexity: Knots, Colourings and Counting, *London Mathematical Society Lecture Note Series 186*, Cambridge University Press, 1993.

[9] 山下雅史: 捜索問題 - 移動する対象を探索する, 離散構造とアルゴリズム III (室田一雄編), 近代科学社, pp.115-162, 1998.