単体分割を利用した結び目の近傍抜き取りプログラムの試作

A Prototype Program for Sampling the Neighborhood of a Knot

http://www.tani.cs.chhs.nihon-u.ac.jp/g-2005/sone/

谷 研究室　曽根 英則
Hidenori SONE

概要

ダイアグラムで表現されている結び目を３次元空間上の四面体分割に埋め込み，結び目の近傍を抜き取り，表示をするトポロジスト支援ツールを作成.

1　はじめに

結び目とは１本の紐の両端を結んだものである．これを数学的に厳密に定義するならば３次元ユークリッド空間内に埋め込まれた頂点が有限個である１本の単純閉折線である．結び目理論はトポロジー（位相幾何学）とよばれる研究分野の１つである．トポロジーとはある変形を幾何学的対象にほどこしても変わらないような性質を研究する学問である．ここでは結び目が伸縮自在のゴムでできていると考える．本研究の目的は３次元空間上における単体分割でできる多様体から結び目の近傍を抜き取った多様体を表示するトポロジスト支援ツールを $C++$ のクラスライブラリの $LEDA$ を使って実装することである．このツールの作成の理由は単体分割から結び目の近傍を抜き取った多様体や結び目の近傍がトポロジストの興味の対象であり，また有用なものだからである．結び目の近傍の抜き取りの方法として一般的に以下のような方法が知られている．

1. 四面体分割でできる四面体の頂点から結び目を生成
2. 各四面体に対し重心細分を行う
3. 結び目の近傍を抜き取る
4. 結果を表示
5. 結び目の近傍を抜き取る

結び目は３次元上で定義されるが一般にトポロジスト（トポロジーの研究者）は結び目を考えるときにはダイアグラムで考えるので，先にダイアグラムとして表現されている結び目を入力し単体分割をして結び目の近傍の抜き取りを行う方法の方が結び目の形が自由に決められ，有用であると思ったのでこの方法での実装も考えたのだが完成にはいたらなかった.

2　準備

2.1　結び目について

結び目 ($KNOT$) とは直感的に言えばループしている一本の紐，つまり一本の紐の両端を結んだものである．

厳密に言えば３次元ユークリッド空間内に埋め込まれた頂点が有限個である１本の単純閉折線である（単純とは自分自身と交わらないということである）．結び目は３次元でのみ定義されるので２次元で結び目を考えるときは**ダイアグラム**で考えることが多い．ダイアグラムとは結び目を平面に射影した**正則射影図**に結び目の交差の上下の情報を加えたものである．

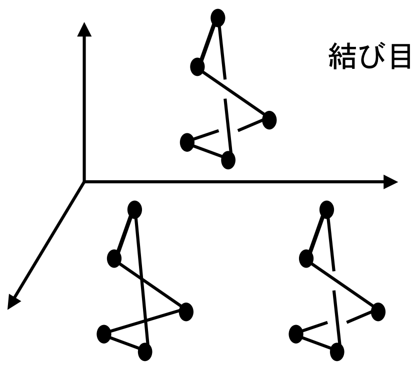

図 0: 結び目とダイアグラム

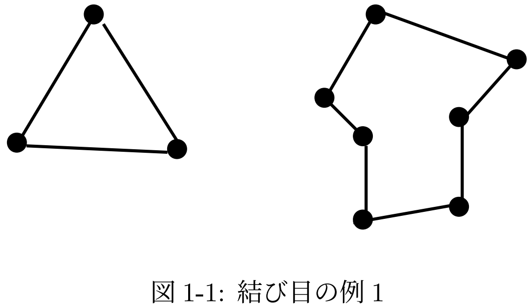

図 1-1: 結び目の例 1

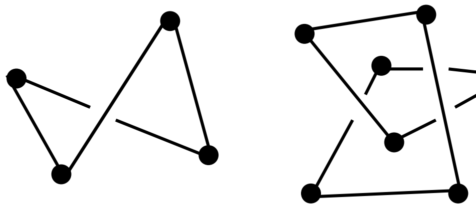

図 1-2: 結び目の例 2

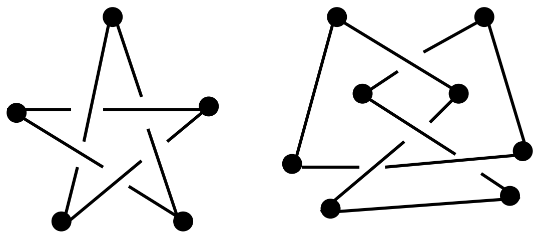

図 1-3: 結び目の例 3

結び目とトポロジー

結び目というのは**トポロジー**（位相幾何学）で考えられる．トポロジーとはある変形を幾何学的対象にほどこしても変わらないような性質を研究する学問である．ここでは結び目がゴムでできていると考えてもらえばよい．つまりゴムであるので伸ばしたり縮めたりしてもよいが切ったり，貼り付けたりすることは許されない．

このように結び目をゴムとして考えるので図 1-1，図 1-2 の結び目はすべて同じである．図 1-1，図 1-2 のように紐の両端をただ結んだだけのような結び目（円周となる紐）を**自明な結び目**と言う．しかし図 3 の結び目は自明な結び目とは言わない．これはハサミで切ってつなぎ直さないかぎり自明な結び目にはならないからである．

結び目の数学的な定義

有限個の点 $v_0, v_1, ..., v_n$ において

K:KNOT（結び目）

$K = K(v_0, v_1, ..., v_{n-1}, v_n) = v_0v_1 \cup v_1v_2 \cup ... \cup v_{n-1}v_n$

ただし $v_0 = v_n$ かつ $n \geq 3$ かつ単純

$v_0, v_1, ..., v_n$ を K の頂点，各線分 $v_0, v_1, ..., v_{n-1}, v_n$ を K の辺とする．

単純であるとはどの 2 つの辺もそれら共通の端点以外に共通点を持たないことである．

2.2 単体分割について

単体とは 0 次元では点，1 次元では線分，2 次元では三角形，3 次元では四面体である（4 次元以上でも単体は定義できる）．

一般的に 2 次元における単体分割を三角形分割，3 次元における単体分割を四面体分割と言う．

三角形分割とは 2 次元において頂点集合が与えられたときその頂点集合によってできる凸包を三角形に分割することである．また四面体分割は 3 次元において頂点集合が与えられたときその頂点集合によってできる凸包を四面体に分割することである（凸包：すべての頂点を含む最小の凸多角形，及び凸多面体）．

単体分割の数学的な定義

$S = \{p_1, p_2, ..., p_n\}$ 頂点集合

$CH(S)$： S の凸包 （S を含む最小の凸多角形，凸多面体）

d 次元における単体分割を T としたとき T は以下のように表せる．

$T = \{S_1, S_2, ...S_m\}$

$\{S_i \quad (i = 1, 2, ..., m) | S_i \subset S\}$

ただし以下の条件を満たすものとする．

1. $\dim CH(S_i) = d$

   $|S_i| = d + 1$
2. $\cup_{i=1}^{m} CH(S_i) = CH(S)$
3. $i \neq j$ で $CH(S_i) \cap CH(S_j) = \emptyset$

   or $CH(S_i) \cap CH(S_j) =$ 頂点 or 辺を共有

ドロネーの三角形分割について

三角形分割において 1 番良い分割の仕方は，分割によってできる三角形がすべて正三角形になることである．

ドロネーの三角形分割は分割によってできる三角形の最大角を最小に，最小角を最大に，辺長の和を最大に（つまり正三角形に近い三角形に）する三角形分割である．ドロネーの三角形分割であるなら分割された三角形二つでできる凸四角形において一方の三角形の外接円が他の三角形の頂点を内部に含むことはない．

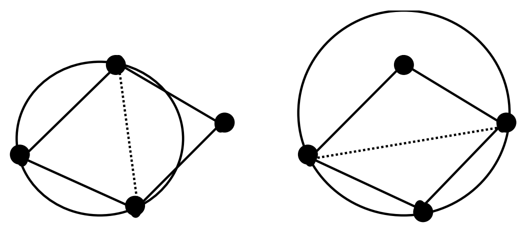

図 2: ドロネーの三角形分割の例（左）

ドロネーの四面体分割について

同様に 3 次元において 1 番良い四面体分割は分割してできる四面体がすべて正四面体であることで，この四面体分割の四面体をなるべく正四面体に近い状態にする四面体分割をドロネーの四面体分割という．

ドロネーの四面体分割であるならば分割された四面体二つでできる凸多面体において一方の四面体の外接球が他の四面体の頂点を内部に含むことはない．

単体分割, ドロネーの単体分割の構成法

任意の三角形分割構成法

S: 2 次元上の n 個の頂点集合 ($n \geq 3$)

1. 頂点 p を x 座標順に添加していく
2. 頂点 p を添加するとき既に添加されている頂点の凸包への接線を求める
3. 2 本の接線にはさまれる凸包の境界上の点をすべて p と結ぶ

1, 2, 3 の作業を繰り返す

ドロネーの三角形分割構成法

S: 2 次元上の n 個の頂点集合 ($n \geq 3$)

1. S の任意の三角形分割を構成する.
2. S の凸包上の辺をのぞいた三角形分割の辺に対してもしその辺に隣接する 2 つの三角形の和が凸四角形であり, かつドロネーの三角形分割でない（三角形二つでできる凸四角形において一方の三角形の外接円が他の三角形の頂点を内部に含むことはない）.
3. 凸四角形の対角線を交換する.
4. 違う辺に対する三角形の和についても考え 2, 3 の作業を繰り返す

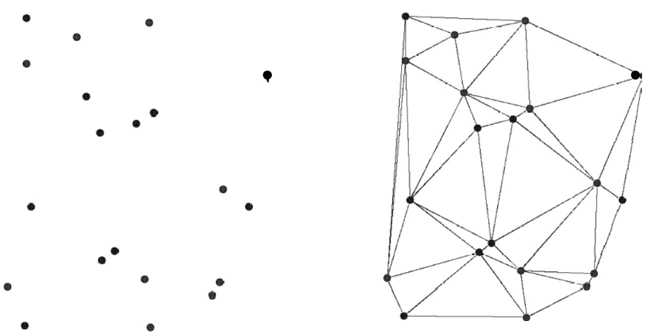

図 3: 入力頂点（左）, 入力頂点に対する三角形分割（右）

任意の四面体分割構成法

S: 3 次元上の n 個の頂点集合 ($n \geq 4$)

1. 頂点 p を x 座標順に添加していく
2. 頂点 p を添加するとき既に添加されている頂点の凸包への接線を求める
3. 2 本の接線にはさまれる凸包の境界上の点をすべて p と結ぶ

1, 2, 3 の作業を繰り返す

ドロネーの四面体分割構成法

S: 3 次元上の n 個の頂点集合 ($n \geq 4$)

1. S の任意の四面体分割を構成する.
2. S の凸包上の辺をのぞいた四面体分割の辺に対してもしその辺に隣接する 2 つの四面体の和が凸多面体であり, かつドロネーの四面体分割でない（四面体二つでできる凸多面体において一方の四面体の外接球が他の四面体の頂点を内部に含むことはない）.
3. 凸多面体の対角線を交換する.
4. 違う辺に対する四面体の和についても考え 2, 3 の作業を繰り返す

2.3 重心細分

重心とは重さの中心を意味する. 図形における重心は各頂点に同重量がかかったとしたときのバランスがとれる点である.

単体（1 次元では線分, 2 次元では三角形, 3 次元では四面体）を分けることで新たな単体を作り出すことを細分という. そのなかでも分けられた単体すべてが元の単体の重心を頂点に持つように細分することを重心細分という. そして重心細分を 2 回行うとは 1 回目の重心細分によってできるすべての単体に対して重心細分を行うことである.

なお本研究においては 2 次元での単体の重心細分は重心, 各辺の中点, 各頂点でできる単体（三角形）に分け, 3 次元での重心細分は四面体の重心, 各面（三角形）の重心, 各辺の中点, 各頂点でできる単体（四面体）に分けている.

重心細分の例

$\triangle ABC$ において辺 BC の中点を M_1, 辺 AB の中点を M_2, 辺 AC の中点を M_3 として辺 AM_1 と辺 CM_2 と辺 BM_3 の交わる点が重心である.

（また同様に $\triangle ABC$ において辺 BC の中点を M_1 として辺 AM_1 を $2:1$ にする点が重心である.）

$\triangle ABC$ の重心, 中点, 頂点によって 6 つの三角形に分けることができる, これが三角形の重心細分である.

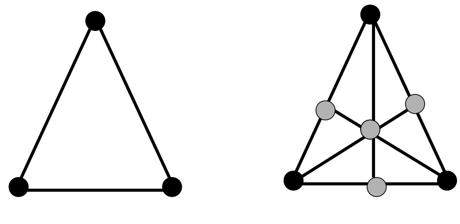

図 4: 三角形の重心細分

四面体の重心細分は四面体 $OABC$ において $\triangle ABC$ の重心を g_1 同様に他の三角形の重心 g_2, g_3, g_4 としたとき重心に向かい合う頂点とを辺でつなぎその交点が四面体 $OABC$ の重心 G である. （四面体 $OABC$ において $\triangle ABC$ の重心 g_1 と頂点 O を結んだ辺 Og_1 を 3:1 にする点が四面体 $OABC$ の重心である.）

そして四面体の重心 G, 四面体の各面の三角形の重心 g_i, 中点, 頂点によって 24 個の四面体に分けることができる，これが四面体の重心細分である．

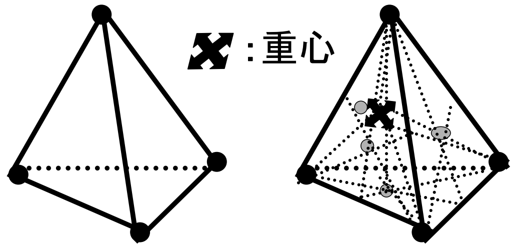

図 5: 四面体の重心細分

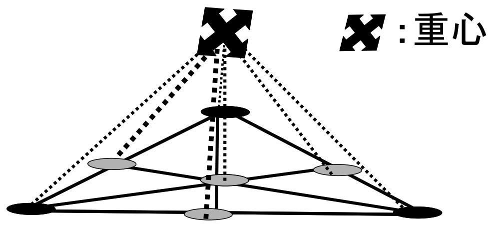

図 6: １つの面に対してできる四面体６個

2.4 結び目の近傍の抜き取り

三角形分割を使っての結び目近傍の抜き取り

２次元においては，入力を頂点集合とし三角形分割を行う．そして三角形分割によってできる三角形の頂点から結び目の頂点を選び，結び目を生成し，結び目の近傍を定義しそれをを抜き取っていく．この場合の結び目の近傍というのは結び目の頂点と辺に隣接する三角形の集合である（隣接する三角形とは結び目の頂点または辺を共有している三角形である）．

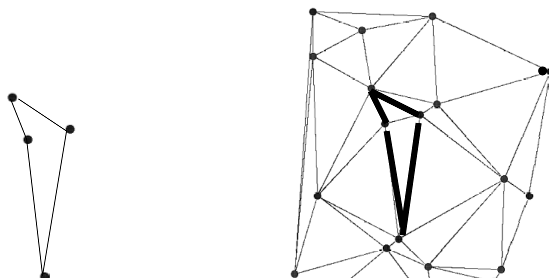

図 7: 結び目と結び目を含む三角形分割

結び目の近傍の数学的な定義

＜２次元における結び目の近傍の数学的な定義＞

T : 三角形分割によってできる三角形の集合
　$(T \subset R^2)$
t_i : T の任意の三角形　　$(\cup t_i = T)$
K : KNOT（入力の結び目）　頂点は有限個
K の近傍を削除
K の近傍 : $N = \{t_i | t_i \in T, K \cap t_i \neq \emptyset\}$
（K の頂点または辺を共有する三角形を削除）

四面体分割を使っての結び目近傍の抜き取り

３次元においては，入力を頂点集合とし四面体分割を行う．そして四面体分割によってできる四面体の頂点から結び目の頂点を選び，結び目を生成し，結び目の近傍を定義しそれをを抜き取っていく．この場合の結び目の近傍というのは結び目の頂点と辺に隣接する四面体の集合である（隣接する四面体とは結び目の頂点または辺を共有している四面体である）．

＜３次元における結び目の近傍の数学的な定義＞

T : 四面体分割によってできる四面体の集合
　$(T \subset R^3)$
t_i : T の任意の四面体　　$(\cup t_i = T)$
K : KNOT（入力の結び目）　頂点は有限個
K の近傍を削除
K の近傍 : $N = \{t_i | t_i \in T, K \cap t_i \neq \emptyset\}$
（K の頂点または辺を共有する四面体を削除）

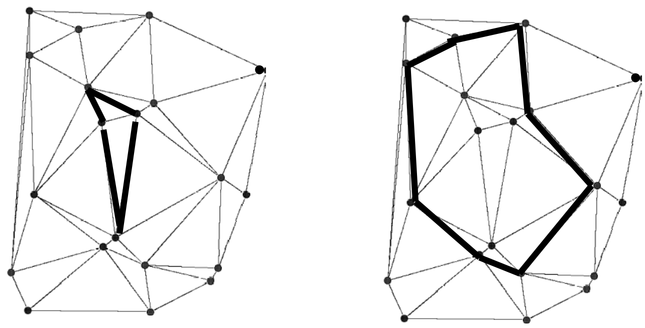

図 8: 結び目 (左)，結び目の近傍

しかしこの方法では図８の左側のように結び目を選んだ場合や例えば結び目の頂点が３つの場合は内側の単体も結び目の近傍として定義されて抜き取ってしまう．これでは結び目の抜き取りではなくなってしまうので内側部分は残さなくてはならない．

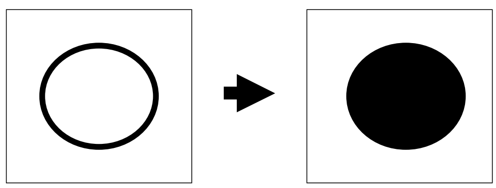

図 9: 内側まで抜き取ってしまう例

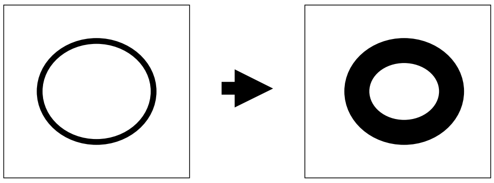

図 10: 厚みをつけて内側は残す

この内側部分を残す方法として単体分割の単体に対し重心細分を２回行い結び目の近傍を抜き取るという方法がある．つまり２次元においては三角形分割の三角形に対し重心細分を２回行い結び目の近傍を抜き取り，３次元においても同様に内側部分まで抜き取ってしまうことがあるので，この場合も四面体に対して重心細分を２回行えばよい．

2.5　重心細分を２回行い結び目の近傍を抜き取る

２次元において重心細分を２回行った後の結び目の近傍の抜き取りをして内側部分を残す方法を結び目の頂点が３つ（三角形）の例で次の図 11 に示す．

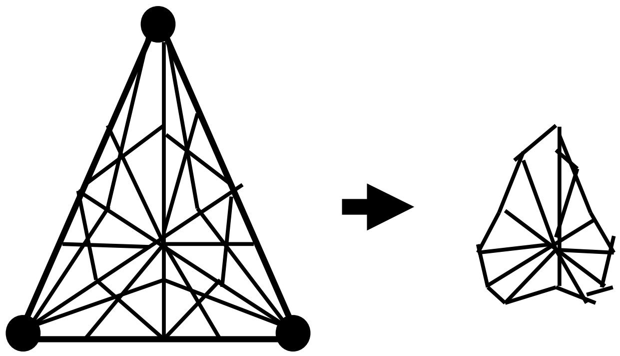

図 11: ２回重心細分後近傍を削除

図から見て取れるように単体に対して２回重心細分を行うことで元の単体の線分に接していない新たな単体を作ることができる，このことから２回重心細分をすることで結び目の内側部分を残すことが可能であることが分かる．図 11 は三角形の例であるが四面体においても同様のことが言える．

3　作成ツールの詳細

3.1　概要

ここで紹介している結び目の近傍の抜き取り方法は，入力である頂点集合に四面体分割を行い，四面体分割でできる四面体の頂点の中からいくつか頂点を選び，結び目とする．そして出力（欲しい結果）となるのは四面体分割の四面体の集合から結び目の近傍を抜き取った多様体である（多様体：局所的にユークリッド空間と同じであるような図形のことである．ここでは特に３次元多様体のことをいい，球や多面体などは（性質は違うが）全て３次元多様体である）．

四面体分割は $LEDA$ に用意されている四面体分割の関数（$D3\_DELAUNAY$）を使用する（これはドロネーの四面体分割である）．ツールのおおまかな流れは以下のようになる．

1. 任意の頂点集合を入力して四面体分割を行う
2. 四面体分割でできる四面体の頂点の中から結び目の頂点を選ぶ
3. 各四面体に２回重心細分
4. 結び目の近傍の抜き取り
5. 結果の表示

なおこの方法が一般的に良く知られている結び目の近傍抜き取りの方法である．

各段階での詳細を以下に記します．

3.2　入力

頂点集合を入力し，四面体分割を行い，四面体分割でできる四面体の頂点の中からいくつか頂点を選び，結び目の頂点とする．ここで注意しなくてはいけないのは結び目の頂点は四面体分割でできる四面体の集合の内側の四面体の頂点から選ぶようにしなくてはいけない．もし外側にあると四面体分割でできる四面体の集合から結び目の近傍を抜き取った多様体が得られないからである．

そして今後の作業のために四面体分割でできる四面体の集合のリストを作る．このリストは各四面体の頂点の座標４つを格納するリストである．

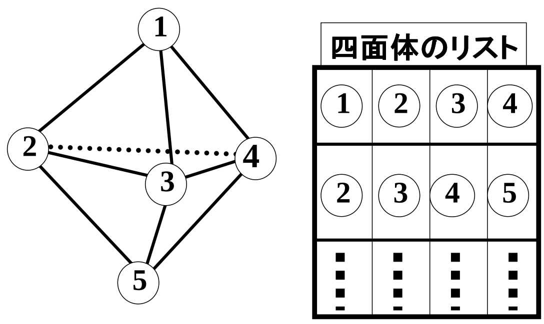

図 12: 四面体のリストの構造

なお頂点の座標を格納する順番などは特に決まっていない．

3.3　重心細分

結び目に接する四面体に対してのみ重心細分

各四面体の辺の中点，各面（三角形）の重心から四面体の重心を求め，重心細分を行い，重心細分によってできた四面体から新たなリストを作成しそれに対して 2.4 章で述べた理由から２回重心細分を行いたいのでもう１度重心細分を行い新たなリストを作る．しかし結び目に接していない外側の四面体に対して２回重心細分を行っても結び目の近傍の抜き取りには関係ないので作成ツールでは四面体分割の後，結び目に接していない外側の四面体などには重心細分を行わないようにしている．また同様に１回目の重心細分で得られた四面体で結び目に接していない四面体には同様に２回目の重心細分は行わないようにしている．

3.4　結び目の近傍の抜き取り

結び目の頂点を共有している四面体リストを削除

結び目の近傍の抜き取りは四面体のリストの中から結び目の頂点を共有しているリストを探しそれを削除する．また２回重心細分を行うことで結び目の辺が重心細分によってできる頂点（結び目の辺の中点，中点によって２分された辺の中点の３つ）で４つに分けられるが，これらの頂点に対しても同様にリストの中から頂点を共有しているものを探し削除する．

しかしこの近傍はすべての四面体のリストから探すのではなく先ほどの結び目に接している四面体にのみ２回重心細分を行ったリストからのみ探せばよい．

3.5　結果の表示（出力）

最後に結び目の近傍を抜き取った四面体のリストと，結び目とは接していなかったので重心細分を行わなかった四面体のリストの情報をもとに結び目の近傍を抜き取った多様体を表示する．なお表示には近傍の抜き取りをした多様体と結び目の近傍の２つを表示するようにしている．

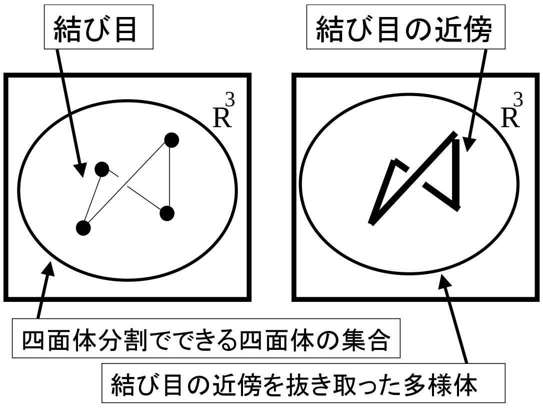

図 13: 入力（左），出力（右）の略図

4　今後の課題

- 入力をダイアグラムとしたときの結び目近傍抜き取りプログラムの実装の方法を考える．
- 現状での作成ツールは結び目の頂点が５つ以上の場合は実行にかなりの時間を要するので実行時間を減らす方法を考える．
- 今回の作成ツールとは違う方法を使って結び目の近傍抜き取りプログラムを実装．

参考文献

[1] 浅野 哲夫 (著)，小保方 幸次 (著)　LEDA で始める C/C++プログラミングー入門からコンピュータ・ジオメトリまで

[2] K. Mehlhorn and St. NaherThe. LEDA Platform of Combinatorial and Geometric Computing

[3] Kurt Mehlhorn (著), Stefan Naher (著) ハードカバー (2000/02/01) Cambridge Univ Pr (Sd)　Leda: A Platform for Combinatorial and Geometric Computing

[4] shingdee choi. The Delaunay Tetrahedorarization from triangulation

[5] J. Hass, J. C. Lagarias and N. pippenger. The computational complexity of knot and links problems. journal of the ACM, Vol. 46, No. 2(1999)185-221

[6] 森元 勘治 (著)，３次元多様体入門

[7] Ian Agol, Joel Hass, William Thurston. 3-manifold knot genus is NP-complete

[8] 譚 学厚 (著), 平田 富夫 (著)　計算幾何学入門一幾何アルゴリズムとその応用

[9] 浅野 哲夫 (著)，計算幾何学