1. はじめに
   1. 結び目理論とは  
       １本の紐を適当に結んで両端をつなぐと，結びのある輪ができる．このような輪を**結び目**という．より厳密に言えば，結び目とは３次元空間内の単純閉曲線である．  
      結び目を柔らかいものでできていると考え，自身を横切らないように３次元空間内で変形させることを**全同位変形**という．２つの結び目が与えられ，一方を全同位変形してもう一方の結び目にできるとき，その２つの結び目は**同じ結び目**であるとみなすことにする．  
      このような状況下で，**どの結び目が同じで，どの結び目が違うのか**を判定する方法を追求するのが結び理論である．

図1.2同じ結び目

＝

図1.1 結び目の作り方

* 1. 結び目を見分けろ  
      図1.2のように，結び目を絵として表現したものを**射影図**という．全く結ばれていない，射影図としてはただの円として表される結び目を**自明な結び目**という．同じ結び目を表す射影図は無数にある．以下は全て自明な結び目の射影図である．  
     以下のような射影図で表される結び目はそれぞれ**三葉結び目**，**８の字結び目**という．  
      さて，現在３つの結び目が登場したが，この中に同じ結び目はあるだろうか？すなわち，全同位変形によっていずれかの結び目をいずれかの結び目に変形できるだろうか？少し考えてみると分かるように，この問いに答えるのはとても難しい．同じ結び目があるのであれば実際に全同位変形をして見せればいいが，問題は**同じでない証明**である．「めちゃくちゃ長い時間紐をいじってみたけど同じなりませんでした」では証明したことにならない．しかし実際には，これら3つの結び目は全て違うということが知られている．それはどのようにして示されたのだろうか．．．

図1.3 自明な結び目

図1.4 三葉結び目

図1.5 ８の字結び目

* 1. 絡み目  
      今までは1本の紐でできた結び目について見てきたが，複数本の紐でも同様のことが考えられる．そこで，いくつかの結び目が絡んでできたものを**絡み目**という．今までと同様，ある絡み目を全同位変形して得られた絡み目は元の絡み目と同じとみなす．  
      絡み目を構成する結び目の数を**成分数**という．結び目は成分数1の絡み目であると言える．成分数2の絡み目の例として**ホップ絡み目**，成分数3の絡み目の例として**ボロミアン環**が挙げられる．n個の自明な結び目からなる絡み目は**n成分の自明な絡み目**という．

図1.6 ホップ絡み目

図1.7 ボロミアン環

* 1. ライデマイスター移動  
      射影図に対して，ライデマイスター移動と呼ばれる3つの変形操作がある．このライデマイスター移動について，**ライデマイスターの定理**という重要な定理が存在する．それは，2つの射影図AとBに対して  
     という定理である．これから見るように，ライデマイスター移動は全同位変形なので（）は当たり前．重要なのは（）である．本論文では証明は省略するが，これが保証されることにより，もし射影図に対して**ライデマイスター移動では変化しない値**が見つかれば，同じ結び目を表す射影図はその値が同じになることが分かる．対偶を取れば，その値が異なる射影図は異なる結び目であると証明できるというわけだ．このような値は**不変量**と呼ばれる．  
      それでは実際にライデマイスター移動がどのようなものか見ていこう．まず**ライデマイスター移動Ⅰ**は以下のようにひねりを作ったり無くしたりする操作である．  
     次に，**ライデマイスター移動Ⅱ**は以下のように隣あっている2辺で交点を作ったり無くしたりする操作である．  
     最後に，**ライデマイスター移動Ⅲ**は以下のように辺と交点に対して，辺を交点の逆側に持っていく操作である．

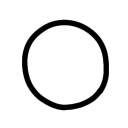
AとBが同じ結び目を表す Aはライデマイスター移動のみでBに変形できる

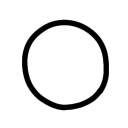
図1.8 ライデマイスター移動Ⅰ

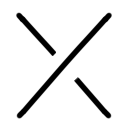
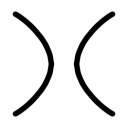
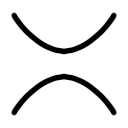
図1.9 ライデマイスター移動Ⅱ

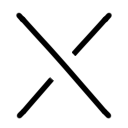
図1.10 ライデマイスター移動Ⅲ

1. Jones多項式
   1. 多項式不変量  
       射影図から計算される多項式で，結び目の不変量になっているものを**多項式不変量**という．本論文で扱う**Jones多項式**がその1つだ．例えば，自明な結び目のJones多項式は「1」である．Jones多項式は不変量なので，自明な結び目のどのような射影図から計算されるJones多項式も「1」になる．つまり，Jones多項式が「1」でない射影図を持つ結び目は自明でないことが証明できる．実際，三葉結び目のJones多項式は「−*t*−4＋*t*−3＋*t*−1」であり，自明な結び目と三葉結び目は異なる結び目であることが証明される．
   2. bracket多項式  
       ここから実際にJones多項式の求め方と，それが不変量になっていることを見ていくが，Jones多項式の前段階としてbracket多項式というものを先に求める必要がある．射影図*L*に対して，そのbracket多項式を＜*L*＞と書くことにする．Bracket多項式は以下の3つのルールで計算される．（多項式の変数は*A*とする．）  
      ルール1はとてもシンプル．自明な射影図のbracket多項式は1である．  
      ルール2はある射影図のbracket多項式を，それよりも簡単な射影図から帰納的に求められるようにするものである．対象の射影図の1つの交点を水平方向と垂直方向に分離させると，交点が少ない2つの新しい射影図ができるが，それらの多項式と元の射影図の多項式の関係を表している．はある交点に着目した対象の射影図を，はその交点を水平方向に分離した以外は元の射影図と全く同じものを，は垂直方向に方向に分離した以外は全く同じものを表す．（2番目の式は1番目の式を90°回転させただけである．）  
      ルール3は成分数に関するルールである．*L*∪は*L*に自明な結び目を交点を持たないように付け加えた射影図を表す．  
       以上の3つのルールだけでどんな射影図のbracket多項式も求めることができる．ルール2で交点数を，ルール3で成分数を減らしていき，最終的に自明な結び目になったところでルール1を使えばよいからである．  
      例として図1.4にある三葉結び目の射影図のbracket多項式を計算してみよう．

ルール1：＜＞＝1

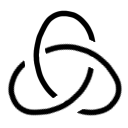
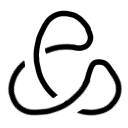
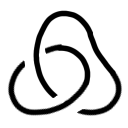
ルール3：＜*L*∪＞＝(−*A*2−*A*−2)＜*L*＞

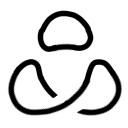
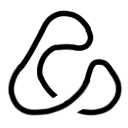
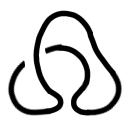
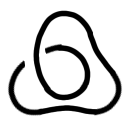
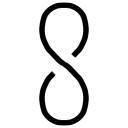
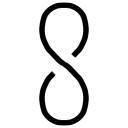
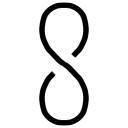
ルール2：＜＞＝*A*＜＞＋*A*−1＜＞

＜＞＝*A*＜図形

AI 生成コンテンツは誤りを含む可能性があります。＞＋*A*−1＜図形

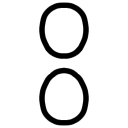
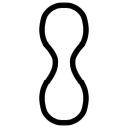
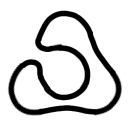
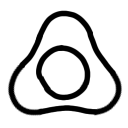
AI 生成コンテンツは誤りを含む可能性があります。＞

＜＞＝*A*＜＞+*A*−1＜＞

＝*A*(*A*＜＞+*A*−1＜＞) +*A*−1(*A*＜＞+*A*−1＜＞)  
＝*A*(*A*(−*A*2−*A*−2)＜＞+*A*−1＜＞) +*A*−1(*A*＜＞+*A*−1＜図形

AI 生成コンテンツは誤りを含む可能性があります。＞)  
＝(−*A*4+1)＜図形

AI 生成コンテンツは誤りを含む可能性があります。＞+ *A*−2＜図形

AI 生成コンテンツは誤りを含む可能性があります。＞  
＝(−*A*4+1)(*A*＜＞+*A*−1＜＞)+ *A*−2(*A*＜＞+*A*−1＜＞)  
＝(−*A*4+1)(*A*(−*A*2−*A*−2)+*A*−1)+ *A*−2(*A*+*A*−1(−*A*2−*A*−2)))  
＝*A*7−*A*3−*A*−5

* 1. *X*多項式  
      ここからはbracket多項式を元に定義される*X*多項式というものについて見ていく．まずは結び目の**向き付け**を考える．結び目の紐をある点から辿っていくことで，結び目や絡み目に向き付けすることができる．このとき各交点は以下のいずれかのようになるが，左のような場合は＋1，右のような場合は−1という符号をつけることにする．  
     向き付けられた射影図*L*に対して，交点の符号の和を**ひねり数**といい，*w*(*L*)と書く．  
     このとき，向き付けられた射影図*L*に対して***X*多項式***X*(*L*)を  
     と定める．図2.2の三葉結び目の*X*多項式は先ほどの例と併せて  
     となる．  
      実はこの*X*多項式，向き付けられた絡み目の不変量となる．このことについてこれから確認していく．絡み目の不変量になっていることを見るには，ライデマイスター移動をしても*X*多項式が変わらないことを見ればよい．まずライデマイスター移動Ⅰについて，のような部分を持つ向き付けられた射影図を*K，K*にライデマイスターⅠを施して図形

     AI 生成コンテンツは誤りを含む可能性があります。の部分がになった射影図を*L*とする．このとき，   
     が成り立つ．よって  
     となるので，このようなライデマイスター移動Ⅰでは*X*多項式は変化しない．他のタイプのライデマイスター移動Ⅰでも同様である．  
      次にライデマイスター移動Ⅱについて，図2.3のように向き付けられた射影図のライデマイスター移動Ⅱで増減する交点の符号は常に＋1と－1のセットになるので，ライデマイスター移動Ⅱでひねり数は不変である．  
     よって，ライデマイスター移動Ⅱでbracket多項式が変化しないことを見れば*X*多項式も変化しないことが分かるが，  
     よりbracket多項式は不変なので，このようなライデマイスター移動Ⅱでは*X*多項式は変化しない．他のタイプのライデマイスター移動Ⅱでも同様である．  
      最後にライデマイスター移動Ⅲについて，ライデマイスター移動Ⅲにおいてもライデマイスター移動Ⅱのときと同様にひねり数は不変である．  
     よって， bracket多項式が変化しないことを見れば*X*多項式も変化しないことが分かるが，ライデマイスター移動Ⅱでbracket多項式が変化しないことから  
     となるのでbracket多項式は不変．よってこのようなライデマイスター移動Ⅲでは*X*多項式は変化しない．他のタイプのライデマイスター移動Ⅲでも同様である．  
      以上より，*X*多項式は向き付けられた絡み目の不変量になっていることが分かった．

(−*A*3)−(−3)(*A*7−*A*3−*A*−5)＝(−*A*9)(*A*7−*A*3−*A*−5)＝−*A*16＋*A*12＋*A*4

*X*(*L*)＝(−*A*3)−*w*(*L*)＜*L*＞

図2.2 ひねり数−3の射影図

図2.1 交点の符号

＋1の交点

−1の交点

*w*(*K*)＝*w*(*L*)−1

*X*(*K*)＝(−*A*3)−*w*(*K*)＜*K*＞

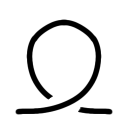
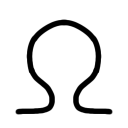
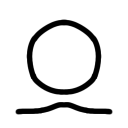
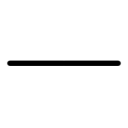
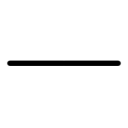
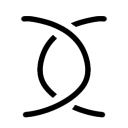
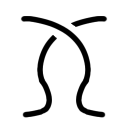
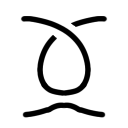
＝(−*A*3)−(*w*(*L*)−1)＜＞  
＝(−*A*3)−*w*(*L*)(−*A*3) (*A*＜＞+*A*−1＜＞)  
＝(−*A*3)−*w*(*L*)(−*A*3) (*A*＜＞+*A*−1(−*A*2−*A*−2)＜＞)  
＝(−*A*3)−*w*(*L*)(−*A*3) (−*A*−3＜*L*＞)  
＝(−*A*3)−*w*(*L*)＜*L*＞  
＝*X*(*L*)

図2.3 ライデマイスター移動Ⅱでひねり数は不変

＋

−

＜＞＝*A*＜＞+*A*−1＜＞

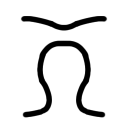
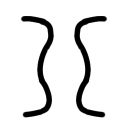
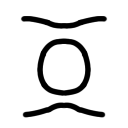
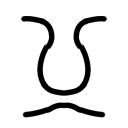
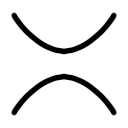
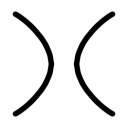
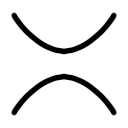
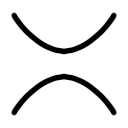
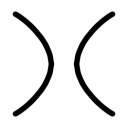
＝*A*(*A*＜＞+*A*−1＜＞)+*A*−1(*A*＜＞+*A*−1＜＞)  
＝*A*2＜＞+＜＞+*A*−1(*A*(−*A*2−*A*−2)＜＞+*A*−1＜＞)  
＝＜＞

図2.4 ライデマイスター移動Ⅲでもひねり数は不変

＋

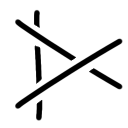
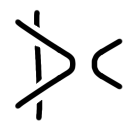
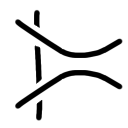
−

＋

＋

＋

−

＜＞＝*A*＜＞+*A*−1＜＞

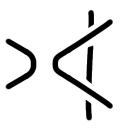
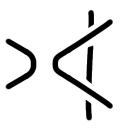
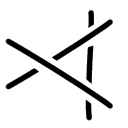
＝*A*＜＞+*A*−1＜＞  
＝＜＞

図2.4 ライデマイスター移動Ⅲでもひねり数は不変

＋

−

＋

＋

＋

−

* 1. Jones多項式  
      向き付けられた絡み目に対して，その*X*多項式に*A*＝*t*−1/4を代入して得られる*t*を変数とする多項式を**Jones多項式**という．*X*多項式やJones多項式は向きの付け方によって変わりうる．例えば以下のように，向き付けの異なる２つのホップ絡み目を考えると  
     左のホップ絡み目のJones多項式は−*t*−1/2−*t*−5/2，右は−*t* 1/2−*t* 5/2となる．これは全同位変形によって左の射影図を右の射影図に向きも含めて一致させることは不可能であることを意味する．しかし結び目においては，向き付けを逆にしても交点と辺の関係は180°回転するだけで交点の符号は変わらない．つまり結び目においては向きに関係なくひねり数が定まるので，Jones多項式も向きに依存せず定まる．  
      先ほどの計算より，三葉結び目のJones多項式は  
     このことから三葉結び目が自明な結び目でないことが分かった．

図2.5 向き付けの異なるホップ絡み目

(−*A*16＋*A*12＋*A*4)| *A*＝*t*−1/4＝−*t*−4＋*t*−3＋*t*−1

1. **Jones多項式 with Python**
   1. 絡み目をコンピュータ語に翻訳する  
       本節では結び目を入力するとそのJones多項式を出力するプログラムを作る方法を紹介する．言語は記述が色々と楽なPythonで行う．Pythonは計算速度が遅いと言われているが，実装してみた感じ特に問題なさそうだった.  
       結び目の射影図をコンピュータで表現する方法を考えよう．n個の交点を持つ射影図に対して，各交点に0〜(n−1)の番号を付ける．（順番はなんでもよい．）また，ある１つの交点に着目すると，そこには４つの辺がある．その辺に0〜3の番号を反時計回りに付ける．（番号0と2を結ぶ辺が上になるように付ける．）こうすることで，全ての辺を(交点番号，辺番号)という組で一意に表すことができる．  
      辺を表せられるようになったので，**どの辺とどの辺が繋がっているか**によって絡み目を表現できるようになる．図3.3のホップ絡み目は   
      と表現できる．

(0,0)-(1,3)　(0,1)-(1,2)　(0,2)-(1,1)　(0,3)-(1,0)

図3.3 (交点番号，辺番号）で全ての辺を表す

(0,2)−

(0,1)

|

−(0,3)

(0,0)

|

|

(1,2)

|

(1,3)

(1,1)−

−(1,0)

図3.2 交点の4つの辺に番号付け

−0

−1

2−

3−

図3.1 交点に番号付け

0

1