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

2004－GI－11 (3)
2004／3／8

単一根有向グラフと対称変換に基づいた囲碁の棋譜管理・閲覧システムの開発

Development of record management / perusal system of the game of go based on Single-Rooted Directed Graph and symmetrical conversion

福羅 勇気 （明治大学）　　玉木 久夫（明治大学）
Yuki Fukura(Meiji University)　　Hisao Tamaki(Meiji University)

Abstract: A game-record manager/brouser for Go is developed which views each collection of game-records as a single-rooted directed graph. The nodes of this graph represent board configurations and the arcs represent moves. Two board configurations are considered identical if one is transformed into the other by a symmetry transformation that maps the board to itself. We describe the basic design of this system, some algorithimic issues in constructing the directed graph form a given collection of records.

1 はじめに

囲碁のゲーム記録である棋譜を管理するソフトウェアは多数存在する [1, 2, 3, 4]。これらのほとんどは個々の棋譜を独立に扱うことを基本としている。棋譜の集まりの中から局面をパターンなどを用いて検索する機能を持ったものも存在するが（例えば [1] など）、あくまで付加的な機能であり、また扱える棋譜の個数も限られている。

これに対して、本稿で述べる棋譜管理・閲覧の方式は次のような考え方に基づいている。棋譜の集まり R は次のような意味で有向グラフ G_R を定義する。G_R のノードは R に現れる局面であり、ノード u からノード v へのアークは R のある棋譜において局面 v が局面 u の直後に現れることを表している。すなわち、アークは棋譜に現れる着手に対応する。ここで、碁盤を碁盤に重ねる８種類の対称変換によって移りあえる局面同士は当然ながら、同一の局面とみなす。囲碁では（特に序盤においてはしばしば）異なる手順で同一局面に至ることがあるので、G_R は一般に木ではない。また、ひとつの棋譜のなかで全く同一の局面が繰り返すこともありうるので、有向閉路を持つこともありうる。本研究では当面置碁を除外して考えるので、G_R は入次数が０の頂点（根）をちょうどひとつ持つ。

概念的には、これまで蓄積された棋譜すべての集まりはこのような意味でひとつの単一根有向グラフを構成している。我々の棋譜管理方式の究極の目標は、この巨大な有向グラフ上をユーザが自由に動き回れるようにすることである。これが可能になれば、棋譜の鑑賞や戦術の研究を行うプロ棋士、アマチュア愛好家、コンピュータ碁の開発者などへの恩恵ははかり知れないものがあるだろう。この目標を達成するためには、棋譜入力や形式変換の実際的な問題は別として、有向グラフの分散表現やデータ交換プロトコルなど解決しなければならない課題は多い。

上の目標への第一歩として有向グラフを主メモリ内に構築する独立型の棋譜管理閲覧システムを開発したので報告する。このシステムでは、現在表示されている局面を含む棋譜名のリストが常時表示され、ユーザはひとつの棋譜の閲覧から局面を共有する他の棋譜への乗り換えを自然かつ容易に行うことができる。なお、本稿ではルールなどの囲碁の初歩的な知識は仮定する [5]。

2 単一根有向グラフ

前節で棋譜の集まりは単一根有向グラフ（SRDG、Single-Rooted Directed Graph) を誘導することを述べた。この節ではSRDGの概念をより詳しく説明し、またそれを表現するデータ構造について述べる。

2.1 単一根有向グラフの例

図2-1に示すSRDGの例が作られる課程を示す。

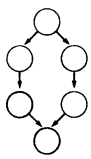

図2-1

図2-2にひとつの棋譜によってSRDGの部分が作られる様子を示す。

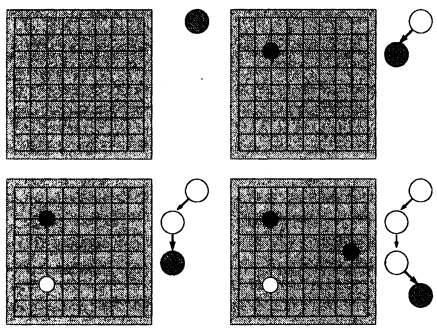

図2-2

図2-3に3手目の結果の局面が同一であるような別の棋譜を入力する様子を示す。すると、同一の局面を表すノードへの入り次数が2となる。

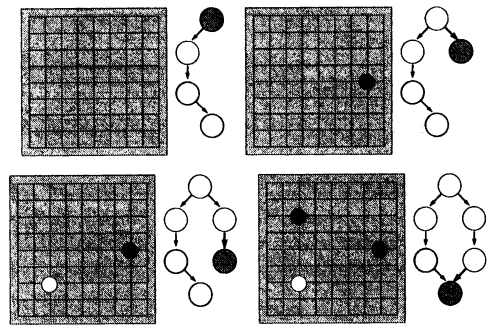

図2-3

2.2 対称変換

碁盤をそれ自身に重ねるような平面上の変換を「対称変換」と言う。通常の盤面の状態に、90°・180°・270°・縦軸・横軸回転、さらに2つの斜軸回転を加えた、総計8通りの対称変換を考える。例えば、図2-4の左の盤面を時計回りに90°回転させると、右の局面と合致する。これらは同一の局面とみなす。

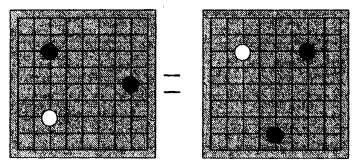

図2-4

2.3 データ構造

局面は論理的には以下の三要素によって決定される。

・盤上の石の配置：各要素が白、黒、空のいずれかを取る $n \times n$ 行列

・手番：白または黒の二値

・コウ情報：コウ取り直後であるかを表す論理値と、それが真実である場合はコウの位置を表す座標

なお、アゲハマ（捕獲された石）の個数はその差だけが問題であり、ゲーム途中でのパスがない限り盤面の石数と手番から決定できるので局面の要素には含めない。

SRDGのノードは局面を表すと述べたが、実際のデータ構造においてはノードの持つ情報は上の論理的な定義における三要素とは異なっている。まず、ノードは盤上の石の配置の情報は持たない。その代わりに、アーク (u,v) にノード u における配置と v における配置の差分の情報を持たせる。SRDGの処理においてはアークを辿って（順方向、逆方向の両方の場合がある）ノードを訪ねていくので、現在訪ねているノードにおける石の配置をこの差分情報を使って維持することができる。また、ノードはその表す局面を含む棋譜名のリストを保持している。これにより、ひとつの棋譜に従って対局を再現していくことや、現在の局面を持つ棋譜名のリストを表示して、ユーザに棋譜の切り替えを許すことなどが可能になる。

3 SRDG構築アルゴリズム

いったんSRDGが主メモリ中に構築されれば、種々の閲覧操作はこのSRDGのアークを辿ることによって行われるため非常に高速である。しかし多数の棋譜に対するSRDGを実用的な時間内に構築することは必ずしも容易ではない。素朴な方法では、数百の棋譜さえも常識的な時間内に処理することができない。本節では、構築時間を短縮するためのアルゴリズム上の工夫を述べ、実験的な比較結果を報告する。

3.1 素朴な方法

既に構築されたSRDGに新たな棋譜を加えることを考える。今その棋譜の途中まで処理されたとし、現在の局面を f、その局面に対応するSRDG中のノードを v_f、処理中の棋譜における f の次の局面を f' とする。v_f の後続ノードのなかに f' を表すノード $v_{f'}$ が存在する場合は、f' を現在局面、$v_{f'}$ を現在ノードとして処理を進めることができる。存在しない場合、f' を表すノードがSRDG中の他の場所にあるかどうかを探さなければならない。素朴な方法では、SRDG中のすべてのパスを局面を再現しながら辿り、f' と比較していく。この方法では、棋譜集合のなかの手数の総数を m としたとき最悪で m^2 のオーダーの構築時間がかかり、実用に耐えないことは明らかである。

3.2 中途での検索の打切1

盤面にN＞1であるN個の石が置かれているとして、局面の比較を行う際には、SRDGをルートから辿っていき、そこから盤面にN個の石が置かれている局面を検索し、同一か否かの判定を行う。しかし検索を行う際に、1手目からN-1手目までについて、その内の一手をどのように対称変換したものも、現在の局面と合致するものがなければ、そこで検索を打ち切ることができる。

例えば、図3-1の「?」における局面が左の盤面のような状態である場合、SRDGをルートから辿っていき、点線のノードにおける局面と比較する。それが同一である場合には、図3-2における左のSRDGのようになり、それが異なる場合には、図3-2における右のSRDGのようになる。

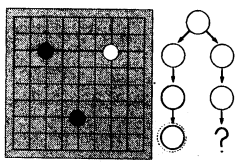

図3-1

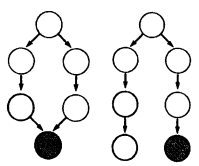

図3-2

しかし図3-3の点線のノードにおける局面が左の盤面のような状態である場合、この黒石をどのよう

に対称変換したものも、図 3-1 の局面の中の黒石とは合致しない。よって、この時点で局面が異なること（図 3-2 における右の SRDG のようになること）が明らかとなり、検索を打ち切ることができる。

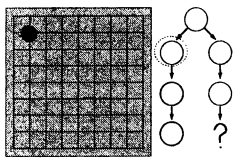

図 3-3

3.3 中途での検索の打切２

先程と同様にして、盤面に N ＞ 1 である N 個の石が置かれているとして、SRDG をルートから辿っていき、盤面に N 個の石が置かれている局面を検索する。その際に、1 手目から N-1 手までについての対称変換を行うが、ここで、８通りの対称変換の内、どれが合致するのかしないのかということを記録しておく。そして、合致しなかった対称変換については、以降の手では検索を行わないようにする。

例えば、図 3-1 の「？」における局面が左の盤面のような状態であり、そして今度は、図 3-4 の点線のノードにおける局面が左の盤面のような状態である場合を考える。このとき、黒石を 270° 回転か縦軸回転させたものが、図 3-1 の中の黒石と合致する。

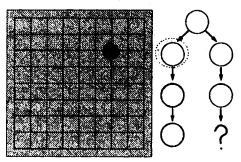

図 3-4

続けて図 3-5 の点線のノードにおける局面が左の盤面のような状態である場合を考える。このときやはり、新たに置いた白石について対称変換を行い、図 3-1 の中の白石と合致するか否かを調べるが、８通りの対称変換の全てを行う必要はなく、一手前の検索で調べて合致した 270° 回転と縦軸回転のみの対称変換を行えば良い。図 3-5 の場合、合致するのは２つの内の縦軸回転の方のみ。このようにして、徐々に検索する幅を狭めていくことができる。

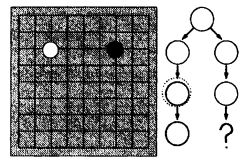

図 3-5

3.4 ハッシュ

ハッシュとは、「キー」と「値」が一組のペアとして関連付けされた配列のことを言う。キーを指定することで、すぐに「値を取り出すことができる。ここでは、局面を数値で表現したものをキーに、それに対応した局面の配列を値にする。局面の数値での表し方は、盤面の中心から一周毎に、黒は５の二乗、白は７の二乗、と値をつけていく。図 3-6 の左図のような局面の場合、$5^3 + 7^3 + 5^4$ で、1093 がこれを表す数値となる。同様にして考えると、図 3-6 の中図も右図の局面を表す数値も 1093 となり、「キー」が同じということになる。これらの局面が、つまりノードが、その「キー」に対応した「値」として配列に記録される。SRDG を作成する際には、まず現在の局面を数値にして、それを「キー」に「値」を取り出す。そのあとは、これまでの方法と同様にして、その「値」が表す局面の中に現在の局面と同じものがあるかどうかを、実際に SRDG を辿って局面を再現しながら検索する。

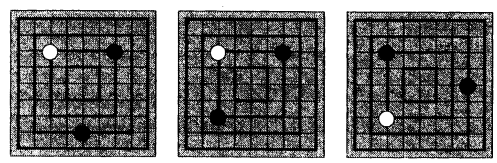

図 3-6

3.5 実験による構築時間の比較

以上のような、構築時間を短縮する方法を取った場合・取らなかった場合の棋譜データの読み込み時間を比較する。比較にはceleron2.3GHz、256MBの計算機を使用した。

以下の５つの場合を考える。

A:構築時間を短縮する方法を何も取らなかった場合

B:中途での検索の打ち切った場合（3.2節）

C:中途での検索の打ち切った場合（3.3節）

D:ハッシュを用いた場合（3.4節）

E:SRDGを構築せずに、棋譜の読み込みだけを行う場合

「置かれる石の数」と「同一局面の数」については、100・300・500局の場合において、それぞれ以下のような数値となり、これらは当然ながら、構築時間を短縮する方法を取るか否かでは変化しない。

表1　置かれる石の数			
対局数	100	300	500
	19916	56969	98590

表2　同一局面の数			
対局数	100	300	500
	476	1352	2526

「読み込み時間」と「辿るノードの数」と「同一局面か否かを調べる回数」は、100・300・500局の場合において、それぞれ以下のような数値になる。

表3　読み込み時間（秒）			
対局数	100	300	500
A	68.187	619.985	2076.782
B	6.609	65.310	315.797
C	4.110	41.469	177.672
D	1.578	4.281	7.672
E	1.187	2.656	4.344

表4　辿るノードの数			
対局数	100	300	500
A	78231065	644465312	1961379514
B	6963561	54739517	175276824
C	2402450	19385018	57138507
D	8145	62915	149143
E	0	0	0

表5　同一局面か否かを較べる回数			
対局数	100	300	500
A	805163	7062566	20317165
B	3411	26749	60303
C	1304	9857	19114
D	821	4940	9653
E	0	0	0

4　部分局面

局面のうち、ある部分だけに着目した記述を「部分局面」と言う。これまで考えてきた全盤の他にも半盤、四半盤、斜半盤があり、全盤を含めた合計４つの部分局面を考える。図４の左端は半盤の例で、盤面の右方に着目したこの例のほかにも、左、上方、下方にそれぞれ着目した半盤があり、これら４つの半盤に対して、ひとつのSRDGを作成する。同様にして、図４の中央は斜半盤の例、右端は四半盤の例であり、４つの四半盤、４つの斜半盤に対しても、それぞれひとつのSRDGを作成し、全盤に対してのSRDGと合わせて、合計４つのSRDGを作る。しかし現時点では、全盤以外のSRDGを作成するプログラムは未完成であり、半盤、四半盤、斜半盤についての同一局面の検索は逐次行うようになっている。

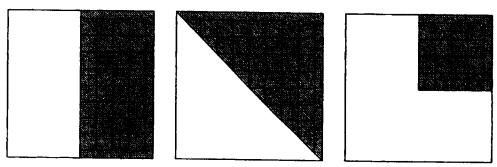

図４

5 ユーザーインターフェイス

ユーザーインターフェイスは図5のようなものになる。

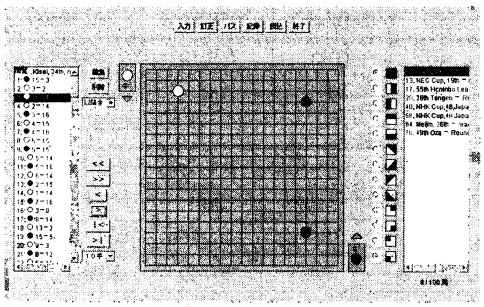

図5

左のリストは現在表示している対局についての情報を、右のリストは現在の局面と同一の局面を持つ棋譜の一覧を、それぞれ表している。左のリストを選択すると、選択した手数のときの状態に盤面が変化し、右のリストを選択すると、選択した対局の表示が行われるようになる。右のリストを選択したときに盤面の状態が変化するか否かは、そのときにどの部分局面を選んでいるかによって分かれる。部分局面の選択は、右のリストの左側にある13の図形を選んで行う。全盤を選んでいるときは、右のリストを選択しても盤面の状態は変化しない。それ以外の部分局面を選んでいるときは、右のリストで選択した対局の何手目から何手目までがその部分局面の状態であるかを検索し、その初めの手数のときの状態に盤面が変化する。

新たに対局を入力する際には、「入力」ボタンを押す。入力した情報を訂正したい場合には「訂正」ボタン、パスを行いたい場合には「パス」ボタン、これまでに入力した対局の情報を保存したい場合には「記録」ボタン、過去に入力した対局の情報を呼び出したい場合には「読込」ボタン、入力を終えたい場合には「終了」ボタンを、それぞれ押す。

モードには編集モードと閲覧モードの2つがある。入力を行っているときは編集モード、入力を行っていて左のリストを選択したときと、読込を行ったときは閲覧モードになる。閲覧モードのときには石を置くことはできない。再び石を置く場合には、「編集」ボタンを押して編集モードに切り替える。

「削除」ボタンを押すことで、選択している対局の一部の削除を行うことができる。また、その下にある6つの移動ボタンを押すことで、局面の移動を行うことができる。

6 まとめ

単一根有向グラフによる棋譜管理構想の第一歩として開発した独立型の棋譜管理閲覧システムの概要を報告した。ネットワーク上の分散表現などは今後の課題である。

参考文献

[1] Great Wall 棋譜管理システム

http://homepage2.nifty.com/yu-ho/

[2] Smart Goban 棋譜管理システム

http://www.din.or.jp/~k_inoue/SmartGo/

[3] Soul Voyager 棋譜管理システム

http://namanama-web.hp.infoseek.co.jp/SVIntro.html

[4] 棋譜松ちゃん 棋譜管理システム

http://www.asahi-net.or.jp/~bq5y-mtt/igo/igosoft.htm

[5] 石倉昇『一人で強くなる囲碁入門　基本を覚えれば上達が早くなる』(日本文芸社、1998年)