

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B1)

(11) 特許番号
特許第4607236号
(P4607236)

(45) 発行日 平成23年1月5日(2011.1.5)

(24) 登録日 平成22年10月15日(2010.10.15)

(51) Int.Cl.

F I

A 6 3 F 13/00 (2006.01)

A 6 3 F 13/00 K

A 6 3 F 3/02 (2006.01)

A 6 3 F 3/02 5 O 1

請求項の数 5 (全 72 頁)

(21) 出願番号	特願2010-50405 (P2010-50405)	(73) 特許権者	504173976
(22) 出願日	平成22年3月8日 (2010.3.8)		株式会社パンダネット
(62) 分割の表示	特願2009-547396 (P2009-547396)		東京都千代田区丸の内三丁目4番1号
	の分割	(74) 代理人	100088214
原出願日	平成21年10月7日 (2009.10.7)		弁理士 生田 哲郎
審査請求日	平成22年3月8日 (2010.3.8)	(74) 代理人	100134588
			弁理士 吉浦 洋一
早期審査対象出願		(72) 発明者	滝 裕子
			東京都千代田区丸の内三丁目4番1号 株
			式会社パンダネット内
		(72) 発明者	佐伯 知哉
			東京都千代田区丸の内三丁目4番1号 株
			式会社パンダネット内
		最終頁に続く	

(54) 【発明の名称】 囲碁の死活判定システム

(57) 【特許請求の範囲】

【請求項 1】

コンピュータで囲碁の死活判定を行う、囲碁の死活判定システムであって、
前記死活判定システムは、
対局者による入力を受け付ける入力受付部と、
判定対象の石または目標石の死活判定を行う死活判定処理部と、
前記死活判定処理部における死活判定の結果を出力する出力処理部と、を備えており、
前記死活判定処理部は、
予め定められた判定領域内であって、前記判定領域の外部に隣接する点のうち少なくとも一以上の点に、前記判定対象の石と異なる色の石を配置する外部処理部と、
前記配置した後の盤面の状態において死活判定処理を行う死活判定実行部と、を有することを特徴とする囲碁の死活判定システム。

【請求項 2】

コンピュータで囲碁の死活判定を行う、囲碁の死活判定システムであって、
前記死活判定システムは、
対局者による入力を受け付ける入力受付部と、
判定対象の石または目標石の死活判定を行う死活判定処理部と、
前記死活判定処理部における死活判定の結果を出力する出力処理部と、を備えており、
前記死活判定処理部は、
予め定められた判定領域のうち、判定対象の石または目標石を含む閉領域を抽出し、そ

の閉領域の外側であって、外部に隣接する点のうち少なくとも一以上の点に、前記判定対象の石と異なる色の石を配置し、前記新たに配置した石を外部とみなす第1外部処理部と、

前記第1外部処理部で処理がされた後の盤面の状態において死活判定処理を実行する死活判定実行部と、を有する

ことを特徴とする囲碁の死活判定システム。

【請求項3】

前記死活判定処理部は、さらに、

予め定められた判定領域のうち外部のE空点を抽出し、抽出したE空点のうち少なくとも一以上の点に、前記判定対象の石と異なる色の石を配置し、前記新たに配置した石を外部とみなす第2外部処理部、を有しており、

前記死活判定実行部は、

前記第1外部処理部および前記第2外部処理部で処理がされた後の盤面の状態において死活判定処理を実行する、

ことを特徴とする請求項2に記載の囲碁の死活判定システム。

【請求項4】

コンピュータで囲碁の死活判定を行う、囲碁の死活判定システムであって、

前記死活判定システムは、

対局者による入力を受け付ける入力受付部と、

判定対象の石または目標石の死活判定を行う死活判定処理部と、

前記死活判定処理部における死活判定の結果を出力する出力処理部と、を備えており、

前記死活判定処理部は、

予め定められた判定領域のうち外部のE空点を抽出し、抽出したE空点のうち少なくとも一以上の点に、判定対象の石と異なる色の石を配置し、前記新たに配置した石を外部とみなす第2外部処理部と、

前記第2外部処理部で処理がされた後の盤面の状態において死活判定処理を実行する死活判定実行部と、を有する

ことを特徴とする囲碁の死活判定システム。

【請求項5】

コンピュータを、

対局者による入力を受け付ける入力受付部、

判定対象の石または目標石の死活判定を行う死活判定処理部、

前記死活判定処理部における死活判定の結果を出力する出力処理部、として機能させる囲碁の死活判定プログラムであって、

前記死活判定処理部は、

予め定められた判定領域内であって、前記判定領域の外部に隣接する点のうち少なくとも一以上の点に、前記判定対象の石と異なる色の石を配置し、

前記配置した後の盤面の状態において死活判定処理を行う、

ことを特徴とする囲碁の死活判定プログラム。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、コンピュータ上で囲碁の対局ゲームを行う場合の、囲碁の死活判定システムに関する。

【背景技術】

【0002】

コンピュータ上でコンピュータを相手に、あるいはネットワークを介してほかのユーザを相手に、囲碁の対局ゲームを行うことが出来る。囲碁は陣地の広さを争うゲームであるが、ある局面において自らの石を活かすためにはどうしたらよいのか、あるいは相手の石を殺すためにはどうしたらよいのか、を常に考慮しながら戦略を立案し、着手を行うこと

10

20

30

40

50

が重要である。

【 0 0 0 3 】

例えば、自らの石が死んでいることを早期に判断できれば、当該陣地を放棄し、別の場所に集中を行うことが出来、全体的な戦略としても非常に効率的である。そのため、自らの石、あるいは相手の石が活着しているのか死んでいるのかを判断することは、囲碁においては極めて重要である。なお石が活着しているのか死んでいるのかを「死活」という。また活着している石とは、相手方の着手により取られない石または取られても新たに相手方に取りられない石を生じうる石のことであり、死んでいる石とは、活石以外であって、相手方の着手により取られる石のことである。

【 0 0 0 4 】

また囲碁のゲーム中の戦略のほかにも、終局状態においても石の死活を判断しなくてはならず、その観点からも石の死活は重要である。そのため詰碁と呼ばれる、部分的な死活を問う問題もある。

【 0 0 0 5 】

このように、囲碁において、石の死活を判定することは極めて重要であるが、実際に石の死活を判定することは容易ではない。特にコンピュータが自動的に行うことは難しい。そこで、例えば下記特許文献、非特許文献にあるように石の死活にかかる処理を行うシステムが開示されている。

【 先行技術文献 】

【 特許文献 】

【 0 0 0 6 】

【 特許文献 1 】 特開平 6 - 1 2 6 0 2 1 号公報

【 特許文献 2 】 特開平 1 0 - 2 7 2 2 5 9 号公報

【 特許文献 3 】 特開平 1 1 - 2 2 6 2 6 2 号公報

【 特許文献 4 】 特開 2 0 0 2 - 2 9 2 1 2 6 号公報

【 非特許文献 】

【 0 0 0 7 】

【 非特許文献 1 】 株式会社パングネット、" 実戦詰碁「死活ナビ」"、[online]、インターネット<URL : <http://www.pandanet.co.jp/members/lifedeath/index.htm>>

【 非特許文献 2 】 トーマス・ウルフ (Thomas Wolf)、" G o T o o l s "、[online]、インターネット<URL : <http://lie.math.brocku.ca/gotools/>>

【 非特許文献 3 】 " 詰碁「鈍」"、[online]、インターネット<URL : <http://hp.vector.co.jp/authors/VA024708/>>

【 非特許文献 4 】 日本コンピュータ囲碁協会、" 詰碁プログラム "、[online]、インターネット<URL : <http://www.geocities.jp/otto4yoshi/jcga/indexkp.html>>

【 発明の概要 】

【 発明が解決しようとする課題 】

【 0 0 0 8 】

特許文献 1 に記載のシステムでは石の死活に係る処理を行っているものの、それは、対局者が死んでいる石を指定する際に、一つの石を指定することで、当該石に隣接する一群の石も死んでいることを指定することが出来るシステムである。従って、対局者は一つの石を指定するだけで、その周辺の石を指定することなく、一群の石を死んでいる石として指定することが出来るので、対局者が死んでいる石を指定するプロセスを減らすことは出来る。しかし、石の死活そのものは対局者が判断しなくてはならず、また死活を自動的に判定できるシステムではない。

【 0 0 0 9 】

また特許文献 2 に記載のシステムでは、碁盤に置かれている石やその特徴を管理することが出来る。例えば碁盤に置かれている石同士のつながりや一群を構成する石などを管理することが出来る。従って、碁盤の石やその並びの管理などを行うとともに、非常に単純なパターン（たとえば 2 眼あれば活き）の石の活きの判定を行うことは出来る。しかし、

10

20

30

40

50

非常に単純なパターンの判定しか行えず、実際の対局での使用に耐えうるものではない。

【 0 0 1 0 】

特許文献 3 に記載のシステムでは、対局の途中で、盤面上にすでに置かれている石が囲碁的に強いが弱いかを判定することは出来る。囲碁的に石が強い、弱いとは、ある石の外側が囲われていればその石は弱く、外側が囲われていなければその石は強いことを意味する。このシステムの場合、ある石の周囲（ここでは 4 目以内）に置かれている異なる色の石の数によって、当該石の強弱を判定している。しかし、石が囲碁的に強いが弱いかを判定できるだけであり、石の死活判定そのものが行われるわけではない。

【 0 0 1 1 】

さらに特許文献 4 に記載のシステムは、囲碁の終局時に石の死活判定を自動的に行うシステムである。このシステムは終局時、すなわち陣地（閉領域）がすでに出来ていることが前提となり、そのうえで簡易的に死活の判定を行っているに過ぎない。従って、終局していない状態、すなわち対局中に死活の判定を行うことは出来ないし、また終局時であっても、限られたパターンの死活の判定しか行うことは出来ない。

【 0 0 1 2 】

以上のように、上述の各特許文献に記載の死活に係る処理では、かならずしも優れた死活判定を行うことが出来ない。

【 0 0 1 3 】

また上述の各特許文献に記載の死活判定を行うシステムのほかにも、非特許文献に記載のように、コンピュータ上で石の死活判定を行うシステムがある。例えば、問題図と正解図、想定される失敗図を何パターンも事前に記憶させておき、それらとマッチングさせながら判定を行う方法である。

【 0 0 1 4 】

この場合、想定されていない着手が行われた場合には判定を行うことが出来ないし、そもそも問題図と正解図、想定される失敗図のパターンを何パターンも記憶させておくことは、記憶量の問題からあまり好ましいものではない。

【 0 0 1 5 】

理論的には死活判定を行うには総あたりを行えば可能である。しかし死活判定の処理には、一般的に $O(n!)$ の処理数が必要とされている（ n は死活判定処理を行う領域の大きさ）。そして死活判定の場合、その結果、すなわち石が生きるか死ぬかを早く知ることが所望される。しかし、総あたりの場合、上述のような処理数が必要となるので、単純に総あたりを行ったのでは、その結果を知るまでに時間がかかることも多く、対局者はストレスを感じてしまう。

【課題を解決するための手段】

【 0 0 1 6 】

そこで本発明者は、従来知られている死活判定システムよりも、高速に死活判定の結果を出力することが可能な囲碁の死活判定システムを発明した。

【 0 0 1 7 】

第 1 の発明は、コンピュータで囲碁の死活判定を行う、囲碁の死活判定システムであって、前記死活判定システムは、対局者による入力を受け付ける入力受付部と、判定対象の石または目標石の死活判定を行う死活判定処理部と、前記死活判定処理部における死活判定の結果を出力する出力処理部と、を備えており、前記死活判定処理部は、予め定められた判定領域内であって、前記判定領域の外部に隣接する点のうち少なくとも一以上の点に、前記判定対象の石と異なる色の石を配置する外部処理部と、前記配置した後の盤面の状態において死活判定処理を行う死活判定実行部と、を有する囲碁の死活判定システムである。

【 0 0 1 8 】

本発明のように構成することで、判定領域内の空点に相手側の石を適宜配置することが可能となる。これによって、死活判定処理の際の判定対象となる点が少なくなる。その結果、単純に死活判定処理を実行するよりも高速に死活判定処理の結果を出力させることが

10

20

30

40

50

可能となる。

【0019】

第2の発明は、以下のように構成することも出来る。すなわち、コンピュータで囲碁の死活判定を行う、囲碁の死活判定システムであって、前記死活判定システムは、対局者による入力を受け付ける入力受付部と、判定対象の石または目標石の死活判定を行う死活判定処理部と、前記死活判定処理部における死活判定の結果を出力する出力処理部と、を備えており、前記死活判定処理部は、予め定められた判定領域のうち、判定対象の石または目標石を含む閉領域を抽出し、その閉領域の外側であって、外部に隣接する点のうち少なくとも一以上の点に、前記判定対象の石と異なる色の石を配置し、前記新たに配置した石を外部とみなす第1外部処理部と、前記第1外部処理部で処理がされた後の盤面の状態において死活判定処理を実行する死活判定実行部と、を有する囲碁の死活判定システムである。

10

【0021】

上述の発明において、以下のような構成をさらに備えていても良い。すなわち、前記死活判定処理部は、さらに、予め定められた判定領域のうち外部のE空点を抽出し、抽出したE空点のうち少なくとも一以上の点に、前記判定対象の石と異なる色の石を配置し、前記新たに配置した石を外部とみなす第2外部処理部、を有しており、前記死活判定実行部は、前記第1外部処理部および前記第2外部処理部で処理がされた後の盤面の状態において死活判定処理を実行する、囲碁の死活判定システムのように構成することも出来る。

【0022】

20

第3の発明は、以下のように構成することも出来る。すなわち、コンピュータで囲碁の死活判定を行う、囲碁の死活判定システムであって、前記死活判定システムは、対局者による入力を受け付ける入力受付部と、判定対象の石または目標石の死活判定を行う死活判定処理部と、前記死活判定処理部における死活判定の結果を出力する出力処理部と、を備えており、前記死活判定処理部は、予め定められた判定領域のうち外部のE空点を抽出し、抽出したE空点のうち少なくとも一以上の点に、判定対象の石と異なる色の石を配置し、前記新たに配置した石を外部とみなす第2外部処理部と、前記第2外部処理部で処理がされた後の盤面の状態において死活判定処理を実行する死活判定実行部と、を有する囲碁の死活判定システムである。

【0023】

30

以上の各発明のように構成しても第1の発明と同様の技術的効果を得ることが可能となる。

【0024】

本発明のプログラムをコンピュータに読み込ませて実行することで、上述の発明を構成することも出来る。すなわち、コンピュータを、対局者による入力を受け付ける入力受付部、判定対象の石または目標石の死活判定を行う死活判定処理部、前記死活判定処理部における死活判定の結果を出力する出力処理部、として機能させる囲碁の死活判定プログラムであって、前記死活判定処理部は、予め定められた判定領域内であって、前記判定領域の外部に隣接する点のうち少なくとも一以上の点に、前記判定対象の石と異なる色の石を配置し、前記配置した後の盤面の状態において死活判定処理を行う、囲碁の死活判定プログラムのように構成しても良い。

40

【発明の効果】

【0025】

本発明の囲碁の死活判定システムを用いることによって、対局中、終局時に限られず、また従来の死活判定システムよりも高速に死活判定の結果を出力することが出来る。

【図面の簡単な説明】

【0026】

【図1】囲碁の対局をネットワークを介して実現する場合の構成の一例を模式的に示す図である。

【図2】本発明の全体の処理機能の一例を模式的に示す概念図である。

50

【図 3】本発明で用いられるサーバやコンピュータ端末のハードウェア構成の一例を模式的に示す図である。

【図 4】実施例 1 における死活判定処理の処理機能の一例を模式的に示す概念図である。

【図 5】実施例 1 における死活判定処理の処理プロセスの一例を模式的に示すフローチャートである。

【図 6】実施例 1 における対局画面の一例である。

【図 7】実施例 1 における編集画面の一例である。

【図 8】実施例 1 における編集画面の一例である。

【図 9】実施例 1 の盤面の状態の一例である。

【図 10】実施例 1 の盤面の状態の一例である。

10

【図 11】実施例 1 の盤面の状態の一例である。

【図 12】実施例 1 における結果の画面の一例である。

【図 13】実施例 1 における全体の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 14】実施例 1 における処理 A の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 15】実施例 1 における処理 B の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 16】実施例 1 における処理 C の詳細な処理プロセスの一例を模式的に示すフローチャートである。

20

【図 17】実施例 1 における処理 D の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 18】E 点を示す図である。

【図 19】複数の石の群における E 点を求める処理を示す図である。

【図 20】O 点を示す図である。

【図 21】複数の石の群における O 点を求める処理を示す図である。

【図 22】E 空点、O 空点の一例を示す図である。

【図 23】実施例 2 における死活判定処理の処理機能の一例を模式的に示す概念図である。

【図 24】実施例 2 における死活判定処理の処理プロセスの一例を模式的に示すフローチャートである。

30

【図 25】実施例 2 における対局画面の一例である。

【図 26】実施例 2 における編集画面の一例である。

【図 27】実施例 2 における編集画面の一例である。

【図 28】実施例 2 の盤面の状態の一例である。

【図 29】実施例 2 の盤面の状態の一例である。

【図 30】実施例 2 の盤面の状態の一例である。

【図 31】実施例 2 の盤面の状態の一例である。

【図 32】実施例 2 の盤面の状態の一例である。

【図 33】実施例 2 の盤面の状態の一例である。

40

【図 34】実施例 2 の盤面の状態の一例である。

【図 35】実施例 2 の盤面の状態の一例である。

【図 36】実施例 2 の盤面の状態の一例である。

【図 37】実施例 2 における対局画面のほかの一例である。

【図 38】実施例 2 の盤面の状態のほかの一例である。

【図 39】実施例 2 の盤面の状態のほかの一例である。

【図 40】実施例 2 の盤面の状態のほかの一例である。

【図 41】実施例 2 の盤面の状態のほかの一例である。

【図 42】実施例 2 の盤面の状態のほかの一例である。

【図 43】実施例 2 の盤面の状態のほかの一例である。

50

【図 4 4】実施例 2 の盤面の状態のほかの一例である。

【図 4 5】実施例 2 の盤面の状態のほかの一例である。

【図 4 6】実施例 2 の盤面の状態のほかの一例である。

【図 4 7】実施例 2 における全体の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 4 8】実施例 2 における処理 A の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 4 9】実施例 2 における処理 A 2 の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 5 0】実施例 2 における処理 A 3 の詳細な処理プロセスの一例を模式的に示すフローチャートである。

10

【図 5 1】実施例 2 における処理 A 4 の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 5 2】実施例 2 における処理 B の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 5 3】実施例 2 における処理 B 2 の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 5 4】実施例 3 における死活判定処理の処理機能の一例を模式的に示す概念図である。

【図 5 5】実施例 3 における死活判定処理の処理プロセスの一例を模式的に示すフローチャートである。

20

【図 5 6】0 眼への変形処理の一例を示す図である。

【図 5 7】後手 1 眼への変形処理の一例を示す図である。

【図 5 8】1 眼への変形処理の一例を示す図である。

【図 5 9】実施例 3 における対局画面の一例である。

【図 6 0】実施例 3 における編集画面の一例である。

【図 6 1】実施例 3 における編集画面の一例である。

【図 6 2】実施例 3 の盤面の状態の一例である。

【図 6 3】実施例 3 の盤面の状態の一例である。

【図 6 4】実施例 3 の盤面の状態の一例である。

30

【図 6 5】実施例 3 の盤面の状態の一例である。

【図 6 6】実施例 3 の盤面の状態の一例である。

【図 6 7】実施例 3 における全体の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 6 8】実施例 3 における処理 D 1 の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 6 9】実施例 3 における処理 D 2 の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 7 0】実施例 3 における処理 A の詳細な処理プロセスの一例を模式的に示すフローチャートである。

40

【図 7 1】実施例 3 における処理 B、処理 C、処理 B'、処理 C' の詳細な処理プロセスの一例を模式的に示すフローチャートである。

【図 7 2】実施例 4 における死活判定処理の処理機能の一例を模式的に示す概念図である。

【図 7 3】実施例 4 における死活判定処理の処理プロセスの一例を模式的に示すフローチャートである。

【図 7 4】実施例 4 における対局画面の一例である。

【図 7 5】実施例 4 における編集画面の一例である。

【図 7 6】実施例 4 における編集画面の一例である。

【図 7 7】実施例 4 の盤面の状態の一例である。

50

- 【図 7 8】実施例 4 の盤面の状態の一例である。
- 【図 7 9】実施例 4 の盤面の状態の一例である。
- 【図 8 0】実施例 4 の盤面の状態の一例である。
- 【図 8 1】実施例 4 の盤面の状態の一例である。
- 【図 8 2】実施例 4 の盤面の状態の一例である。
- 【図 8 3】実施例 4 の盤面の状態の一例である。
- 【図 8 4】実施例 4 の盤面の状態の一例である。
- 【図 8 5】実施例 4 の盤面の状態の一例である。
- 【図 8 6】実施例 4 の盤面の状態の一例である。
- 【図 8 7】実施例 4 の盤面の状態の一例である。 10
- 【図 8 8】実施例 4 の盤面の状態の一例である。
- 【図 8 9】実施例 4 の盤面の状態の一例である。
- 【図 9 0】実施例 4 の盤面の状態の一例である。
- 【図 9 1】実施例 4 の盤面の状態の一例である。
- 【図 9 2】実施例 4 の盤面の状態の一例である。
- 【図 9 3】実施例 4 の盤面の状態の一例である。
- 【図 9 4】実施例 4 の盤面の状態の一例である。
- 【図 9 5】実施例 4 の盤面の状態の一例である。
- 【図 9 6】実施例 4 の盤面の状態の一例である。
- 【図 9 7】実施例 4 の盤面の状態の一例である。 20
- 【図 9 8】実施例 4 の盤面の状態の一例である。
- 【図 9 9】実施例 4 の盤面の状態の一例である。
- 【図 1 0 0】実施例 4 の盤面の状態の一例である。
- 【図 1 0 1】実施例 4 の盤面の状態の一例である。
- 【図 1 0 2】実施例 4 の盤面の状態の一例である。
- 【図 1 0 3】実施例 4 の盤面の状態の一例である。
- 【図 1 0 4】実施例 4 の盤面の状態の一例である。
- 【図 1 0 5】実施例 4 における全体（処理 A）の詳細な処理プロセスの一例を模式的に示すフローチャートである。
- 【図 1 0 6】実施例 4 における処理 B の詳細な処理プロセスの一例を模式的に示すフロー 30
- チャートである。
- 【図 1 0 7】実施例 4 における処理 D の詳細な処理プロセスの一例を模式的に示すフロー
- チャートである。
- 【図 1 0 8】実施例 4 における処理 C の詳細な処理プロセスの一例を模式的に示すフロー
- チャートである。
- 【図 1 0 9】実施例 4 における処理 C 2 の詳細な処理プロセスの一例を模式的に示すフロー
- チャートである。
- 【図 1 1 0】実施例 4 における処理 C 3 の詳細な処理プロセスの一例を模式的に示すフロー
- チャートである。
- 【図 1 1 1】実施例 4 における処理 E の詳細な処理プロセスの一例を模式的に示すフロー 40
- チャートである。
- 【図 1 1 2】実施例 4 における処理 F の詳細な処理プロセスの一例を模式的に示すフロー
- チャートである。
- 【図 1 1 3】実施例 4 における処理 F 2 の詳細な処理プロセスの一例を模式的に示すフロー
- チャートである。
- 【図 1 1 4】実施例 4 における処理 F 3 の詳細な処理プロセスの一例を模式的に示すフロー
- チャートである。
- 【図 1 1 5】実施例 4 における処理 G の詳細な処理プロセスの一例を模式的に示すフロー
- チャートである。
- 【図 1 1 6】実施例 4 における処理 H の詳細な処理プロセスの一例を模式的に示すフロー 50

チャートである。

【発明を実施するための形態】

【0027】

本発明の囲碁の死活判定システム1（以下、単に「死活判定システム1」という）は、インターネットなどのネットワークを介した囲碁の対局システムにおいて、あるいはネットワークを介さない囲碁の対局システムにおいて、用いることが出来る。なお、本明細書では対局システムと同時に、死活判定システム1を用いる場合を説明するが、死活判定システム1単独で機能させることも可能である。その場合にも同様に処理が行える。

【0028】

図1に、本発明の死活判定システム1を備えた、ネットワークを介した囲碁の対局システムの構成の一例を模式的に示す。この場合、各対局者が利用するコンピュータ端末（対局者端末3）と、対局システムを実現するサーバ2とがネットワークを介して情報の送受信が可能である。本明細書では、対局システムと死活判定システム1とは同一のサーバ2内で処理が実行される場合を示すが、異なるサーバ2で処理が実行されていても良い。また上述のように、サーバ2で対局システムが実現されるほか、死活判定システム1を備えた対局システムが対局者端末3に備えられることで、ネットワークを介さずに本発明に係る処理が実行されても良い。また、対局システムに本発明の死活判定システム1がその機能として組み込まれ、一体的に構成されていても良い。

10

【0029】

本発明の囲碁の死活判定システム1の処理機能の一例の概念図を模式的に図2に示す。上述のように、本明細書ではネットワークを介して囲碁の対局システムが構成される場合なので、サーバ2に各処理機能が備えられ、対局者が利用する対局者端末3との間で情報の送受信をネットワークを介して行うことで、その処理が実現される。

20

【0030】

またサーバ2および対局者端末3であるコンピュータ端末のハードウェア構成の一例を図3に模式的に示す。サーバ2およびコンピュータ端末は、プログラムの演算処理を実行するCPUなどの演算装置10と、情報を記憶するRAMやハードディスクなどの記憶装置11と、ディスプレイ（画面）などの表示装置12と、キーボードやポインティングデバイス（マウスやテンキーなど）などの入力装置13と、演算装置10の処理結果や記憶装置11に記憶する情報をインターネットやLANなどのネットワークを介して送受信する通信装置14とを有している。サーバ2およびコンピュータ端末上で実現する各機能（各手段）は、その処理を実行する手段（プログラムやモジュールなど）が演算装置10に読み込まれることでその処理が実行される。各機能は、記憶装置11に記憶した情報をその処理において使用する場合には、該当する情報を当該記憶装置11から読み出し、読み出した情報を適宜、演算装置10における処理に用いる。また、コンピュータ端末、サーバ2は、複数のコンピュータ端末またはサーバ2に、その機能が分散配置されていても良い。

30

【0031】

本発明における各手段は、その機能が論理的に区別されているのみであって、物理上あるいは事実上は同一の領域を為していても良い。

40

【0032】

サーバ2は入力受付部20と死活判定処理部21と出力処理部22とを有する。なお本明細書では図示していないが、同一または異なるサーバ2には、囲碁の対局システムを実現する処理機能が備えられており、そこでは盤面の情報（すなわち、どの目にどの石が置かれているか、手番など、囲碁の対局をコンピュータで実現するために必要な情報）を記憶・管理している。そして死活判定処理システムが死活判定処理を行う場合には、その盤面の情報などを適宜、対局システムから取得し、死活判定処理に用いる。

【0033】

入力受付部20は、死活判定を行うことの要求や着手の情報、手番の情報、判定対象の石、目標石などの情報などの死活判定を行うのに必要な情報を対局者端末3から受け付け

50

る。また出力処理部 22 は、死活判定処理部 21 における処理結果や対局者端末 3 に表示させる情報などを、対局者端末 3 に送る。

【0034】

死活判定処理部 21 は、入力受付部 20 で死活判定処理を行うことの要求を対局者端末 3 から受け付けると、対局システムから盤面の情報を取得し、その盤面または対局者によって編集された編集後の盤面の情報に基づいて死活判定処理を行う。またこの際に、盤面すべてについてではなく、どの領域について死活判定処理を行うかの領域を指定する情報を対局者端末 3 から受け付けることで、当該領域についての盤面の情報を取得し、死活判定処理を行っても良い。また、そのほか手番の情報、判定対象の石、目標石などの死活判定処理に必要な情報を対局者端末 3 から入力受付部 20 が受け付け、それを用いて処理を実行する。

10

【0035】

死活判定処理は基本的には、従来と同様に、石が置かれていない点に総あたりで石を置き、それをすべてのパターンについて逐次判定することで、石の死活判定処理を行うが、その場合、処理時間が多く必要となる。そこで死活判定処理部 21 は、上記総あたり処理の前に、後述する各処理の一以上を任意に実行することで総あたり処理の高速化を実現する。なお死活判定処理部 21 における死活判定処理としては、従来と同様の死活判定処理プログラムを用いることが出来る。なお以下の各実施例では、総あたりによる死活判定処理を行う場合を示すが、それ以外の従来の死活判定処理プログラムによる方法を用いることも可能である。また総あたりでの死活判定処理とは、従来の死活判定処理プログラムにおいて用いられている死活判定処理であっても良く、必ずしも判定領域内のすべての空点に対する総あたり処理を行わなくても良い。例えば判定領域内の空点のうち、一部の空点については処理を省略するなどの方法であっても良い。例えば、価値が全く同じ着手が複数あった場合、一つの着手のみを実行する、といったように処理を省略する方法などがある。すなわち、死活判定実行部 214 は、任意の方法により、判定領域内にある判定対象の石または目標石に対しての死活判定処理を実行できる方法であれば如何なる方法であっても良い。また本明細書での死活判定処理として総あたり（または一部の空点について処理を行わない効率化処理を施した総あたり）での死活判定処理の場合を示すが、総あたりでの死活判定処理でなくても、死活を判定できる方法であれば、如何なる方法を用いることも出来る。

20

30

【0036】

なお、従来の死活判定処理プログラムの一例として、上述の各非特許文献に記載のコンピュータプログラムなどがあり、それらを適宜用いることが可能であるが、それらに限定されるものではない。

【0037】

なお、後述の各実施例の説明で用いる用語を以下に説明する。判定対象の石とは、生きまたは死にの結論を出すべき石であり、判定領域内の方の色の石のすべてが含まれる。少なくとも一つの石が生きる場合、死活判定の結果は生きとなり、すべての石が取られる場合、死活判定の結果は死となる。また目標石とは判定対象の石の一部の石またはすべての石であって、生きまたは死にの結論を出すべき石である。ただし目標石が指定された場合、目標石のすべてが生きる場合、死活判定の結果は生きとなり、目標石の少なくとも一つの石が取られる場合、死活判定の結果は死となる。なお目標石は対局者によって指定されても指定されなくても良い。

40

【0038】

判定領域とは、判定の際、石の着手を可能とする領域である。グループとは隣接（縦横に位置する石）して連結している同色の石の集まりである。外部とは、判定の際、着手を不可能とする領域である。この領域にある石のグループは生き石として扱われ、また一つのグループが判定領域と外部の両方に含まれることはない。

【実施例 1】

【0039】

50

上述の死活判定処理部 2 1 a における高速処理の第 1 の実施例を以下に説明する。本実施例における死活判定処理部 2 1 a の処理機能の一例の概念図を図 4 に示す。本実施例の死活判定処理部 2 1 a では、総あたりの死活判定処理を行うにあたり、死活判定の結果が出る可能性が高い点から優先的に総あたりを行うように、各点の処理の優先度を変更した上で、その優先度に従って、総あたりの死活判定処理を実行する場合を示す。

【 0 0 4 0 】

本実施例の死活判定処理部 2 1 a では初期判定処理部 2 1 0 と第 1 判定処理部 2 1 1 と第 2 判定処理部 2 1 2 と優先度変更処理部 2 1 3 と死活判定実行部 2 1 4 と優先度記憶部 2 1 5 とを備える。

【 0 0 4 1 】

初期判定処理部 2 1 0 は、本実施例における高速判定が可能であるかを判定する。すなわち、判定領域の内部にある、本来の判定対象の石の一部または全部を外部とし、その外部より内側にある、本来の判定対象の石とは異なる色の石を内部とした判定が可能であるかを判定する。判定可能であると判定した場合、本来の判定対象の石に隣接する、異なる色の石を目標石として設定する。

【 0 0 4 2 】

第 1 判定処理部 2 1 1 は、初期判定処理部 2 1 0 で高速判定可能であると判定すると、判定領域内の判定対象の石と異なる色の石を内部、本来の判定対象の石を外部とし、本来の判定対象の石に隣接する相手方の石を目標石とした死活判定を実行する。なお、目標石として、本来の判定対象の石に隣接する相手方の石の一部、あるいは判定領域内にある相手方の石の一部または全部を目標石として死活判定を実行しても良い。

【 0 0 4 3 】

第 2 判定処理部 2 1 2 は、第 1 判定処理部 2 1 1 で生き（勝ち）となる点がなかった場合、判定領域内の判定対象の石と異なる色の石を内部、本来の判定対象の石を外部とし、死活判定を実行する。

【 0 0 4 4 】

なお第 1 判定処理部 2 1 1 において、本来の判定対象の石に隣接する相手方の石の全部を目標石とした死活判定を実行した結果、生き（勝ち）となる点がなかった場合に、本来の判定対象の石に隣接する相手方の石の一部、あるいは判定領域内にある相手方の石の一部または全部を目標石として死活判定をさらに実行しても良い。そしてこの死活判定処理でも生き（勝ち）となる点がなかった場合に、第 2 判定処理部 2 1 2 における死活判定処理を実行するように構成することも出来る。

【 0 0 4 5 】

優先度変更処理部 2 1 3 は、第 1 判定処理部 2 1 1、第 2 判定処理部 2 1 2 で生きとなる点があった場合に、その点について、後述する優先度記憶部 2 1 5 に記憶する各点の優先度を上げる処理を行う。また第 1 判定処理部 2 1 1 で生きとなる点があった場合のその着点以外の点、第 2 判定処理部 2 1 2 で生きとなる点があった場合のその着点以外の点、あるいは第 2 判定処理部 2 1 2 で生きとなる点がなかった場合の判定領域内の点について、後述する優先度記憶部 2 1 5 に記憶する各点の優先度を下げる処理を行う。

【 0 0 4 6 】

死活判定実行部 2 1 4 は、第 1 判定処理部 2 1 1、第 2 判定処理部 2 1 2 でそれぞれ設定した条件に従って、各判定領域内の空点に対する死活判定処理を実行する。この死活判定処理は従来と同様の死活判定処理プログラムを用いることが出来る。なお以下の各実施例では、総あたりによる死活判定処理を行う場合を示すが、それ以外の従来 of 死活判定処理プログラムによる方法を用いることも可能である。また総あたりでの死活判定処理とは、従来 of 死活判定処理プログラムにおいて用いられている死活判定処理であっても良く、必ずしも判定領域内のすべての空点に対する総あたり処理を行わなくても良い。例えば判定領域内の空点のうち、一部の空点については処理を省略するなどの方法であっても良い。例えば、価値が全く同じ着手が複数あった場合、一つの着手のみを実行する、といったように処理を省略する方法などがある。すなわち、死活判定実行部 2 1 4 は、任意の方法

10

20

30

40

50

により、判定領域内にある判定対象の石または目標石に対しての死活判定処理を実行できる方法であれば如何なる方法であっても良い。

【 0 0 4 7 】

優先度記憶部 2 1 5 は、各点についての着手処理の優先度を記憶している。

【 0 0 4 8 】

次に本実施例における死活判定処理の処理プロセスの一例を図 5 のフローチャートを用いて説明する。

【 0 0 4 9 】

対局者が利用する対局者端末 3 同士でネットワークを介して囲碁の対局が行われている。この際に囲碁の対局は対局システムを実現するサーバ 2 上でその処理が行われている。そしてその対局中、あるいは終局後、対局者のいずれかがその表示される画面から所定のボタン等を押下することで、死活判定処理を行うことの要求が対局者端末 3 から、死活判定処理を行うサーバ 2 に送信される。対局画面の一例を図 6 に示す。

10

【 0 0 5 0 】

対局画面が図 6 の場合、その画面のうち、「死活ナビ」など、死活判定処理を行うことを要求するボタンを押下することで、死活判定処理を行うことの要求が対局者端末 3 からサーバ 2 に送信される。

【 0 0 5 1 】

サーバ 2 の入力受付部 2 0 でその要求を受け付けると、死活判定処理部 2 1 a は、対局システムの記憶装置 1 1 から当該盤面の情報（たとえば棋譜の情報）を取得する。そして盤面の情報が含まれた死活判定処理の画面を出力処理部 2 2 が対局者端末 3 に送信する。この際の画面の一例が図 7 である。なお実際に対局している盤面の情報のほか、すでに対局し、所定の記憶装置 1 1 に保存している棋譜の情報、あるいは観戦している対局の棋譜の情報を死活判定処理部 2 1 a が対局システムの記憶装置 1 1 から取得し、その棋譜の情報を図 7 の画面に含めて対局者端末 3 に送信しても良い。

20

【 0 0 5 2 】

死活判定を行う対局者は、図 7 に示す死活判定処理を行う画面から盤面の編集を行う。例えば表示されている石を指定することで、石を取り除いたり、あるいは石を置く。またマウスなどの入力装置 1 3 をドラッグするなどにより、判定領域を指定する。このように死活判定処理を所望する対局者は、適宜、盤面の情報を編集する。この際の編集される石の情報、手番の情報などは、対局者端末 3 からサーバ 2 に適宜、送られる。

30

【 0 0 5 3 】

上述のように対局者が判定領域を指定することにより、その判定領域内にある結論を出すべき石を判定対象の石として設定される。なお、対局者が判定対象の石のうち、さらに特定の石を結論を出すべき石（目標石）として設定したい場合には、それを目標石として設定することが出来る。判定対象の石のうち少なくとも一つの石が活きる時には死活判定の結論は活きとなり、すべての石が取られる場合、結論は死となる。なお、目標石を指定した場合には、目標石のすべてが活きる時には死活判定の結論は活きとなり、目標石のうち少なくとも一つが取られる場合、死活判定の結論は死となる。なお目標石の指定は必須ではなく、また目標石を指定するかしないかではその処理には差がなく、活き死にの判断基準が変わるだけである。従って、以下の実施例においてはその断りがない限り、対局者が目標石を指定しない場合を説明する。

40

【 0 0 5 4 】

編集後の死活判定処理を行う画面の一例を図 8 に示す。なお図 8 では判定対象の石が黒の 1 3 子、手番は白であるとする。囲碁において「手番」とは次の順番の石の色を示すほか、手番から見た勝ち負け（活き死に）を判定することを示す。従って、この場合、次の順番は白石であり、さらに白石から見て黒の 1 1 子の死活判定を行うことを意味している。

【 0 0 5 5 】

例えば図 8 の画面における「判定処理」を押下することで、編集後の盤面の情報（棋譜

50

の情報と判定対象の石の情報など)と手番の情報と、死活判定処理を開始することの要求とが対局者端末3からサーバ2に送信される。サーバ2の入力受付部20で編集後の盤面の情報と手番の情報と死活判定処理を開始することの要求を受け付けると、死活判定処理部21aは、入力を受け付けた編集後の盤面の情報に基づいて、死活判定処理を開始する。

【0056】

まず死活判定処理部21aは、死活判定処理を開始するにあたり、各点(石の置かれていない交点、すなわち空点)の処理の優先度を任意に、仮に設定する(S100)。ここで仮に設定された各点の優先度は、優先度記憶部215に記憶される。この各点の処理の優先度は、どのような方法で設定しても良い。この優先度は、どの空点から処理を実行するかを特定する優先度であって、順位付けされていても良いし、各空点ごとにポイントが付加され、そのポイントが高い順(あるいは低い順)に処理が行われるように優先度が設定されていても良い。なおこの各点の優先度の設定は、従来と同様である。

10

【0057】

次に、死活判定処理部21aの初期判定処理部210は、本実施例における高速処理が実行可能かを、盤面の情報(編集後の盤面の情報)に基づいて判定する(S110)。すなわち、判定領域の内部にある、本来の判定対象の石を外部とし、その外部より内側にある、本来の判定対象の石とは異なる色の石を内部とした判定が可能であるかを判定する。また、この際に初期判定処理部210は、本来の判定対象の石に隣接している石(判定対象の石と隣接している相手方の石)をすべて目標石として特定する。

20

【0058】

つまり盤面の情報は図8の状態であり、本来の判定対象の石が黒石の13子であるので、初期判定処理部210は、判定領域内の白石を内部、判定対象の黒石を外部とし、目標石として星によって示される白石を特定する(この盤面の状態を模式的に示すのが図9である)。そして図8の場合、判定対象の石である黒石の内部に白石が含まれていることから、本実施例の高速判定が可能であると判定する(S120)。

【0059】

なおS110の判定処理において、本来の判定対象の石の領域内に相手方の石がないなどの理由で高速判定できない場合には、本実施例による処理を中止し、従来どおり、総あたりによる死活判定処理を開始する(S130)。

30

【0060】

また、S110の判定処理において、上述のように高速判定が可能であることを初期判定処理部210が判定すると、第1判定処理部211は、判定対象の石の内部を判定領域とした死活判定処理を、現在の手番(すなわちここでは白の手番)で行う。この死活判定処理は、S110で設定した判定領域内のみに対しての総あたりを行う(S140)。すなわち、図10に示すように、白8子を判定対象の石および目標石として、手番白での判定を開始する。従って、図10の白い四角で示されたのが判定領域内の空点であるので、この点に対して、死活判定実行部214が、総あたりで死活判定処理を開始する。なお総あたりの死活判定処理そのものは従来と同様である。

【0061】

そうすると、図10の場合、図11に示すように、白活き(白勝ち)として判定でき、その着点が2カ所のA点であると第1判定処理部211が判定する。

40

【0062】

第1判定処理部211で手番の石についての活き(勝ち)を判定する(活きとなる点がある)と(S150)、その着点について、優先度変更処理部213は、優先度記憶部215に記憶する、S100で設定した仮の優先度から、当該点の優先度をあげ、それ以外の点(第1判定処理部211での判定領域におけるA点以外の点)の優先度を下げる(S160)。例えば着点として判定されたA点の優先順位を最上位(例えば1番と2番)に変更し、順次、順位を繰り下げるとともに、第1判定処理部211での判定領域におけるA点以外の点の優先順位を下げる。あるいはA点についての優先順位を所定順位(例えば

50

10番)だけ繰り上げ、第1判定処理部211での判定領域におけるA点以外の点の優先順位を所定順位だけ繰り下げる。さらに優先度がポイントで管理されている場合には、A点について所定量のポイントを加算し、第1判定処理部211での判定領域におけるA点以外の点について所定量のポイントを減算する。なお優先度の変更処理については、上記に限られず、少なくとも着点の優先度が上がるものであれば如何なるものであっても良い。

【0063】

このように第1判定処理部211で着点として判定された点の優先度を、優先度変更処理部213が優先度が高くなるように変更することで、当該点(着点として判定されたA点)は優先的に処理が行われるようになる。

10

【0064】

また、S140での第1判定処理部211での判定の結果、手番の石についての活き(勝ち)が判定できない(活きとなる点がない)場合には、第2判定処理部212において、S140で指定された目標石を指定せず、S140と同様に、死活判定実行部214で総あたりでの死活判定処理を開始する(S170)。

【0065】

すなわち、S140での第1判定処理部211での判定では、本来の判定対象の石に隣接している石をすべて目標石として指定しているが、第2判定処理部212では目標石を指定せずに、S140と同様に総あたりでの死活判定処理を実行する。従って、図10の場合、S140では白8子を判定対象の石および目標石として手番白での判定を開始したが、S170では、白8子を判定対象の石として手番白での判定を開始する。この場合であっても、総あたりの対象となる空点は、S140とは変わらない。

20

【0066】

そして第2判定処理部212で手番の石についての活き(勝ち)を判定する(活きとなる点がある)と(S180)、その着点について、優先度変更処理部213は、優先度記憶部215に記憶する、S100で設定した仮の優先度から、当該点の優先度をあげ、それ以外の点の優先度を下げる(S190)。また第2判定処理部212で手番の石についての活き(勝ち)を判定出来ない(活きとなる点がない)場合には、優先度変更処理部213は、当該判定領域の各点の優先度を下げる(S200)。

【0067】

30

以上のようにすることで、判定領域の内部にある本来の判定対象の石と異なる色の石を内部、本来の判定対象の石を外部、とした場合の手番の石の活きの判定を行い、その判定結果に基づいて、活きとなる点があるならば、S100で仮に設定した当該着点の優先度を上げ、それ以外の点については優先度を下げることが可能となる。

【0068】

そして以上のような処理を行った後、死活判定処理部21aは、優先度記憶部215に記憶する、第1判定処理部211または第2判定処理部212で変更された、各点の優先度に従って総あたりでの死活判定処理を実行することとなる。

【0069】

第1判定処理部211または第2判定処理部212で活きと判定された際の着点(例えば図11のA点)は、総あたりに死活判定を行う場合にも、ほかの空点よりも優先して処理が行われる。この着点は、本来の判定対象の石の死活判定に影響を与える可能性が高い点(生きるあるいは死ぬと判定できる可能性の高い点)であることから、これらの着点を優先的に処理することで、本来の判定対象の石の死活の結果を早く出す可能性を高めることができる。

40

【0070】

つまり、単純に総あたりする場合、S100で各点ごとに仮に設定された優先度に従って総あたりを実行するため、いつ死活の結果が出るかが分からない。しかし本実施例の死活判定処理部21aにおける処理を行うことで、死活の結果が出る可能性が高い点から優先的に処理するので、死活の結果が早く出る可能性が高くなる。

50

【 0 0 7 1 】

また逆に、第 1 判定処理部 2 1 1 で生きとなる点があった場合の着点以外の点、第 2 判定処理部 2 1 2 で生きとなる点があった場合の着点以外の点、あるいは第 2 判定処理部 2 1 2 で生きとなる点が多かった場合の判定領域の各点の優先度を下げる処理を優先度変更処理部 2 1 3 は行っている、それらの点は、仮に設定された点の処理よりもあとに処理が行われる。これらの点は、本来の判定対象の石の死活判定に影響を与える可能性が低い点（生きる或いは死ぬと判定できる可能性が低い点）であることから、優先度を下げることで、本来の判定対象の石の死活の結果を早く出す可能性を高めることが出来る。

【 0 0 7 2 】

以上のようにして死活判定処理部 2 1 a で死活判定の処理結果を判定すると、出力処理部 2 2 は、その処理結果を対局者端末 3 に送信する。この際に対局者端末 3 で表示される死活判定の結果の画面の一例を図 1 2 に示す。このような画面を対局者が見ることで、死活の結果を知ることが出来る。

10

【 0 0 7 3 】

次に上述の各処理のより詳細な説明を図 1 3 乃至図 1 7 を用いて説明する。なお以下の各実施例の説明で用いるデータ形式および用語を以下に説明する。

【 0 0 7 4 】

盤面データは $Board[x][y]$ とする（ただし $1 \leq x \leq 19, 1 \leq y \leq 19$ ）。そして盤面の各点は空点であれば 0、黒石があれば 1、白石があれば 2 が代入される。

手番は t_n で示される。なお $t_n \wedge 3$ で相手の石の色になるが、「 \wedge 」はビット演算を示す。

20

判定対象の石は $H[x][y]$ とする。判定対象となる石には 1、目標石でもある石には 2 が代入される。

着手の優先順位は $Order[i]$ とする（ただし $0 \leq i < 361$ ）また、 $Order[i].x$ （点 i の x 座標）、 $Order[i].y$ （点 i の y 座標）は座標である。また $Order[i].v$ は変数（点 i の着手の優先順位）である。

判定領域は $Area[x][y]$ とする（ただし $1 \leq x \leq 19, 1 \leq y \leq 19$ ）。判定領域である点には 1 が代入され、判定領域外の点には 0 が代入される。

グループ化とは、ある点から同じ色で、縦または横に隣接している点の集合を配列に代入する処理である。

30

【 0 0 7 5 】

E 点とは、その石から縦路または横路に隣接している（距離が 1 だけ離れた）点の集合である。図 1 8 に示すように、黒 1 子に対する E と書かれている点が E 点となる。また、複数の石の群における E 点は、その石群を構成するすべての石の E 点の和集合から、その石群が存在する点を除いたものとなる。これを示す一例が図 1 9 である。

【 0 0 7 6 】

O 点とは、その石から斜めに隣接している（距離が 1 だけ離れた）点の集合である。図 2 0 に示すように、黒 1 子に対する O と書かれている点が O 点となる。また、複数の石の群における O 点は、その石群を構成するすべての石の O 点の和集合から、その石群が存在する点とその石群の E 点に含まれる点を除いたものとなる。これを示す一例が図 2 1 である。

40

【 0 0 7 7 】

E 空点とは、その石群の E 点のうち、いずれの石も置かれていない点の集合である。O 空点とは、その石群の O 点のうち、いずれの石も置かれていない点の集合である。これらの一例を示すのが図 2 2 である。

【 0 0 7 8 】

まず死活判定処理部 2 1 a は、盤面の情報を読み込み、 $Board[x][y]$ に空点であるか、黒石が置かれているか、白石が置かれているかを記憶していく。また判定領域 $Area[x][y]$ にそれぞれ判定領域であるか否かを記憶していく。また上記のほか、判定対象の石 $H[x][y]$ 、判定領域 $Area[x][y]$ 、手番 t_n など適宜、

50

設定される。そして、総あたり処理を開始するにあたり、空点 ($Board[x][y] = 0$) の点の処理の優先度を任意に、仮に設定する ($S300$)。すなわち $Order[i]$ にそれぞれ優先度を仮に設定していく。ここで設定した各点の優先度は優先度記憶部 215 に記憶される。なお優先度の処理については上述と同様に如何なる方法でも良い。
【0079】

次に、死活判定処理部 21a の初期判定処理部 210 は、初期判定処理 (処理 A) を実行する ($S310$)。

【0080】

具体的には、まず配列 $Area[x][y] = 1$ 、 $Board[x][y] = tn^3$ となる各点の x, y について、 $S3100$ 乃至 $S3120$ の処理を反復する。すなわち、 $Board[x][y]$ を判定対象として判定領域を生成し ($S3100$)、 $S3100$ で設定した判定領域で、外部となる石がすべて、本来の判定の判定対象の石に含まれるかを判定する ($S3110$)。もし含まれない場合には、次の点 (x, y) の処理に移る。もし含まれる場合には、 $S3100$ で設定した領域で判定可能であるかを判定し ($S3120$)、そうでないならば、次の点 (x, y) の処理に移る。また判定可能ならばその時点でループを中断し、第 1 判定処理部 211 における処理に移る。また処理 B に移ることがなかった場合でループを終了した場合には、実施例 1 における高速判定処理を適用することが出来ないで、 $S300$ で設定した仮の優先度 ($Order[i]$ に設定された優先度) に基づいて、総あたりでの死活判定処理を開始させる。

【0081】

以上のように初期判定処理部 210 において初期判定処理 (処理 A) を実行したあと、第 1 判定処理部 211 における第 1 判定処理 (処理 B) を実行する ($S320$)。なお、本来の判定領域が配列 $Area[x][y]$ 、判定対象の石が配列 $H[x][y]$ で示され、さらに、内部についての判定領域が配列 $Area'[x][y]$ 、判定対象の石が配列 $H'[x][y]$ で示される。従って、それぞれの配列 $Area'[x][y]$ 、 $H'[x][y]$ にはそれぞれ値が代入されている。

【0082】

具体的には、まず判定対象の領域について、 $H'[x][y] = 1$ となる各点 (判定対象となる各点) (x, y) について、 $S3200$ 乃至 $S3230$ の処理を反復させる。まず、上記点 (x, y) ($H'[x][y] = 1$ となる点) でグループ化 (各点 (x, y) から、同じ色で縦路または横路につながっている点の集合を配列に代入する) して、それを配列 $G[x][y]$ に代入する ($S3200$)。そして配列 $G[x][y]$ の E 点を求め、配列 $E[x][y]$ に代入する ($S3210$)。

【0083】

そして配列 $E[a][b] = 1$ 、 $Area'[a][b] = 0$ となる点 (a, b) があるかを判定する ($1 \leq a, b \leq 19$) ($S3220$)。すなわち E 点であって、判定領域外の点があるかを判定する。もしないならば、その時点で処理 B を中止する。

【0084】

$S3220$ で上記点があるならば、 $G[c][d] = 1$ となる点 (c, d) で、 $H'[c][d] = 2$ とする ($1 \leq c, d \leq 19$) ($S3230$)。すなわちその点の石を目標石として設定する。

【0085】

以上の処理を $H'[x][y] = 1$ となる各点について行った後、総あたりの死活判定処理を死活判定実行部 214 が、判定領域 $Area'[x][y]$ の各点について行い、生き (勝ち) となる着手点を配列 $R[x][y]$ に代入する ($S3240$)。もし生き (勝ち) となる着手点がない場合には配列 $R[x][y]$ には代入を行わない。なお着手点には、 $R[x][y]$ に 1 が代入され、そうでない点には 0 が代入される。

【0086】

以上のように第 1 判定処理部 211 において第 1 判定処理 (処理 B) を実行したあと、処理 B で生きとなる着手点があるかを判定する ($S330$)。もし生きとなる着手点があ

10

20

30

40

50

る場合（配列 $R[x][y]$ に着手点の代入がある場合）には、後述する優先度変更処理部 213 における処理 D を実行する（S350）。

【0087】

以上のような処理を実行することで、図 5 における S110 乃至 S150 に相当する処理を実行させることが出来る。

【0088】

また S330 において、生きとなる着手点がない場合（配列 $R[x][y]$ に着手点の代入がない場合）には、第 2 判定処理部 212 における第 2 判定処理（処理 C）を実行する（S340）。

【0089】

具体的には、まず判定対象の領域について、 $H'[x][y] = 2$ となる各点（目標石となる点）（ x, y ）について、 $H'[x][y] = 1$ として設定する（S3400）。すなわち目標石となる点を判定対象となる石として設定する。

【0090】

そして、総あたりの死活判定処理を死活判定実行部 214 が、判定領域 $Area'[x][y]$ の各点について行い、生き（勝ち）となる着手点を配列 $R[x][y]$ に代入する（S3410）。もし生き（勝ち）となる着手点がない場合には配列 $R[x][y]$ には代入を行わない。

【0091】

以上のように第 2 判定処理部 212 における第 2 判定処理（処理 C）を実行することで、図 5 における S170 乃至 S180 に相当する処理を実行することが出来る。また、第 2 判定処理部 212 における第 2 判定処理を実行したあと、あるいは S330 において処理 B で生きとなる点があったと判定した場合、優先度変更処理部 213 における優先度変更処理（処理 D）を実行する（S350）。

【0092】

まず、着手の優先度を示す配列 $Order[i]$ （ $0 \leq i < 361$ ）において、 $Order[i].v$ （パラメータである）の最大値を m に代入する（S3500）。

【0093】

そして $i = 0$ から $i < 361$ になるまで、S3510 乃至 S3540 の処理を反復させる。

【0094】

まず $H'[Order[i].x][Order[i].y]$ が 1 以上となる場合には（S3510）、その x, y において、着手点であるか、すなわち $R[Order[i].x][Order[i].y] = 1$ であるかを判定する（S3520）。もし着手点である場合（ $R[Order[i].x][Order[i].y] = 1$ である場合）には、 $Order[i].v$ を $m+1$ だけ増加させる（S3530）。また、着手点ではない場合（ $R[Order[i].x][Order[i].y] = 1$ ではない場合）には、 $Order[i].v$ を $m+1$ だけ減少させる（S3540）。すなわち、着手点である場合には、所定量（ $m+1$ ）だけ優先度を増加させ、着手点ではない場合には、所定量（ $m+1$ ）だけ優先度を減少させる。

【0095】

以上の処理をすべての i について実行したあと、 $Order[i]$ を、その要素 v が大きい順にソート（並べ替え）する（S3550）。これによって、着手点の優先度が高くなり、そうではない点についての優先度が低くなる。

【0096】

このようにして優先度変更処理部 213 での処理が行われると、配列 $Order[i]$ が優先度記憶部 215 に記憶される。以上の処理を実行することで、図 5 における S160、S190 乃至 S200 に相当する処理を実行することが出来る。そしてここで記憶された優先度に基づいて、死活判定実行部 214 が総あたりの死活判定処理を行うことによって、優先度が高い、すなわち着手点である点を優先的に処理することとなる。着手点で

10

20

30

40

50

ある点は死活判定の結果が出やすい点であるので、死活判定の結果の出力を迅速に行うことが出来る。

【 0 0 9 7 】

死活判定実行部 2 1 4 で総あたりの判定処理を実行中に、その死活判定の結果が出た場合には、出力処理部 2 2 がその処理結果を対局者端末 3 に送信する。このような画面を対局者が見ることで、死活の結果を知ることが出来る。

【 0 0 9 8 】

本実施例のように、単純に総あたりの死活判定処理を行う前に、判定領域の内部の相手方の石が活着しているかどうかを判定し、その死活の判定結果をもとに、総あたりの際の着手の優先度を定めることで、判定対象となる石の死活判定処理を高速に実現することが出来る。

10

【実施例 2】

【 0 0 9 9 】

次に、死活判定処理部 2 1 b における高速処理の第 2 の実施例を以下に説明する。本実施例における死活判定処理部 2 1 b の処理機能の一例の概念図を図 2 3 に示す。本実施例の死活判定処理部 2 1 b では、総あたりの死活判定処理を行うにあたり、処理対象となる点を減らした上で、総あたりの死活判定処理を実行する場合を示す。

【 0 1 0 0 】

本実施例の死活判定処理部 2 1 b には、第 1 外部処理部 2 1 6 と第 2 外部処理部 2 1 7 と死活判定実行部 2 1 8 とを備える。

20

【 0 1 0 1 】

第 1 外部処理部 2 1 6 は、本来の判定領域のうち、判定対象の石を含む閉領域を抽出し、その閉領域の外側の点であって所定条件を充足する点に、判定対象の石と異なる色の石を配置し、新たに配置した石を外部とみなすことで、総あたりの処理対象となる点を減らす。

【 0 1 0 2 】

第 2 外部処理部 2 1 7 は、外部の E 空点であって判定領域となる点を抽出し、その各点の E 点に外部の石が 2 以上含まれ、判定対象の石が一つも含まれていない点に、判定対象の石と異なる色の石を配置し、新たに配置した石を外部とみなすことで、総あたりの処理対象となる点を減らす。

30

【 0 1 0 3 】

なお本明細書において、外部とは、判定領域以外の一部または全部の領域にある点である。好ましくは、判定領域以外の点であって、かつ、E 点または O 点が判定領域に含まれる点である。

【 0 1 0 4 】

死活判定実行部 2 1 8 は、第 1 外部処理部 2 1 6、第 2 外部処理部 2 1 7 で処理がされたあとの盤面の状態において、本来の判定領域の空点に対する死活判定処理を実行する。この死活判定処理は従来と同様の死活判定処理プログラムを用いることが出来る。なお以下の実施例では、総あたりによる死活判定処理を行う場合を示すが、それ以外の従来の死活判定処理プログラムによる方法を用いることも可能である。また総あたりでの死活判定処理とは、従来の死活判定処理プログラムにおいて用いられている死活判定処理であっても良く、必ずしも判定領域内のすべての空点に対する総あたり処理を行わなくても良い。例えば判定領域内の空点のうち、一部の空点については処理を省略するなどの方法であっても良い。例えば、価値が全く同じ着手が複数あった場合、一つの着手のみを実行する、といったように処理を省略する方法などがある。すなわち、死活判定実行部 2 1 8 は、任意の方法により、判定領域内にある判定対象の石または目標石に対しての死活判定処理を実行できる方法であれば如何なる方法であっても良い。

40

【 0 1 0 5 】

なお第 1 外部処理部 2 1 6 および第 2 外部処理部 2 1 7 については双方の処理を実行しても良いし、いずれか一つであっても良い。また第 1 外部処理部 2 1 6 および第 2 外部処

50

理部 2 1 7 における処理はその一例であって、それ以外の方法、例えば、判定対象の石と異なる色の石を外部を基準として、新たに配置することで、総あたりの処理対象となる点を減らせれば、如何なる方法であっても良い。

【 0 1 0 6 】

次に本実施例における死活判定処理の処理プロセスの一例を図 2 4 のフローチャートを用いて説明する。

【 0 1 0 7 】

対局者が利用する対局者端末 3 同士でネットワークを介して囲碁の対局が行われている。この際に囲碁の対局は対局システムを実現するサーバ 2 上でその処理が行われている。そしてその対局中、あるいは終局後、対局者のいずれかがその表示される画面から所定のボタン等を押下することで、死活判定処理を行うことの要求が対局者端末 3 から、死活判定処理を行うサーバ 2 に送信される。対局画面の一例を図 2 5 に示す。

10

【 0 1 0 8 】

対局画面が図 2 5 の場合、その画面のうち、「死活ナビ」など、死活判定処理を行うことを要求するボタンを押下することで、死活判定処理を行うことの要求が対局者端末 3 からサーバ 2 に送信される。

【 0 1 0 9 】

サーバ 2 の入力受付部 2 0 でその要求を受け付けると、死活判定処理部 2 1 b は、対局システムの記憶装置 1 1 から当該盤面の情報（たとえば棋譜の情報）を取得する。そして盤面の情報が含まれた死活判定処理の画面を出力処理部 2 2 が対局者端末 3 に送信する。この際の画面の一例が図 2 6 である。なお実際に対局している盤面の情報のほか、すでに対局し、所定の記憶装置 1 1 に保存している棋譜の情報、あるいは観戦している対局の棋譜の情報を死活判定処理部 2 1 b が対局システムの記憶装置 1 1 から取得し、その棋譜の情報を図 2 6 の画面に含めて対局者端末 3 に送信しても良い。

20

【 0 1 1 0 】

死活判定を行う対局者は、図 2 6 に示す死活判定処理を行う画面から盤面の編集を行う。例えば表示されている石を指定することで、石を取り除いたり、あるいは石を置く。またマウスなどの入力装置 1 3 をドラッグするなどにより、判定領域を指定する。このように死活判定処理を所望する対局者は、適宜、盤面の情報を編集する。この際の編集される石の情報、手番の情報などは、対局者端末 3 からサーバ 2 に適宜、送られる。

30

【 0 1 1 1 】

上述のように対局者が判定領域を指定することにより、その判定領域内にある結論を出すべき石を判定対象の石として設定される。なお、対局者が判定対象の石のうち、さらに特定の石を結論を出すべき石（目標石）として設定したい場合には、それを目標石として設定することが出来る。判定対象の石のうち少なくとも一つの石が活きる時には死活判定の結論は生きとなり、すべての石が取られる場合、結論は死となる。なお、目標石を指定した場合には、目標石のすべてが活きる時には死活判定の結論は生きとなり、目標石のうち少なくとも一つが取られる場合、死活判定の結論は死となる。なお目標石の指定は必須ではなく、また目標石を指定するかしないかではその処理には差がなく、生き死にの判断基準が変わるだけである。従って、以下の実施例においてはその断りがない限り、対局者が目標石を指定しない場合を説明するが、目標石を指定した場合には、「判定対象の石」との記載を「目標石」と適宜読み替えることで、その処理が可能となる。

40

【 0 1 1 2 】

編集後の死活判定処理を行う画面の一例を図 2 7 に示す。なお図 2 7 では判定対象の石が黒の 8 子、手番は黒であるとする。従って、この場合、次の順番は黒石であり、さらに黒石から見て黒の 8 子の死活判定を行うことを意味している。

【 0 1 1 3 】

例えば図 2 7 の画面における「判定処理」を押下することで、編集後の盤面の情報（棋譜の情報と判定対象の石の情報など）と手番の情報と、死活判定処理を開始することの要求とが対局者端末 3 からサーバ 2 に送信される。サーバ 2 の入力受付部 2 0 で編集後の盤

50

面の情報と手番の情報と死活判定処理を開始することの要求を受け付けると、死活判定処理部 2 1 b は、入力を受け付けた編集後の盤面の情報に基づいて、死活判定処理を開始する。

【 0 1 1 4 】

まず死活判定処理部 2 1 b の第 1 外部処理部 2 1 6 は、総あたり処理を開始するにあたり、盤面の棋譜の情報を抽出し、判定対象の石を含む閉領域を構成する石の群を抽出する (S 4 0 0)。なお閉領域とは、相手の石を無視して自分の石のみまたは自分の石と外部とで閉じている領域である。

【 0 1 1 5 】

そうすると、図 2 7 の場合、図 2 8 に示すように、二つの閉領域を構成する石の群が抽出できる。図 2 8 (a) は判定対象となる黒石のみによる閉領域であり、図 2 8 (b) は判定対象となる黒石と盤端とによる閉領域である。

【 0 1 1 6 】

そして S 4 0 0 において閉領域がない場合 (S 4 1 0)、第 1 外部処理部 2 1 6 は、その処理を中止し、第 2 外部処理部 2 1 7 における処理を実行する。すなわち、S 4 7 0 に移る。一方、閉領域がある場合 (S 4 1 0)、各閉領域について、閉領域の外部の点であって、閉領域を構成する石の群の E 空点以外で且つ 1 線の O 空点以外の点を抽出する (S 4 2 0)。S 4 2 0 の処理を図 2 8 に対して実行したのが図 2 9 である。図 2 9 において黒四角形で示される点が S 4 2 0 で抽出した点である。

【 0 1 1 7 】

S 4 2 0 の処理を、S 4 0 0 で抽出した各閉領域について処理を実行し (S 4 3 0)、すべての閉領域について S 4 2 0 における点を抽出すると、それらのすべての閉領域で共通する点を抽出する (S 4 4 0)。つまり各閉領域について S 4 2 0 で抽出した点の積となる点を抽出する。これを模式的に示すのが図 3 0 である。

【 0 1 1 8 】

そして S 4 4 0 で抽出した点のうち、判定対象の石とその E 空点を除く、残った点に判定対象の石と異なる色の石 (ここでは白石) を配置する (S 4 5 0)。これを模式的に示すのが図 3 1 である。なお図 3 1 では新たに配置された相手側の石を星印で示している。なおここでは残った点のすべてに相手側の石を配置したが、すべてではなく一部に配置をしても良い。

【 0 1 1 9 】

S 4 5 0 で新たに配置した相手側の石のうち、外部に繋がっていない石を取り除く。そして取り除いた石以外の新たに配置した相手側の石を外部としてみなす (S 4 6 0)。これを模式的に示すのが図 3 2 および図 3 3 である。

【 0 1 2 0 】

以上のようにして第 1 外部処理部 2 1 6 は処理を実行する。そして第 1 外部処理部 2 1 6 における処理が終了後、または第 1 外部処理部 2 1 6 の S 4 0 0 で閉領域がないと判定した場合、第 2 外部処理部 2 1 7 の処理を実行する。

【 0 1 2 1 】

すなわち、第 2 外部処理部 2 1 7 は、判定領域内にある外部の E 空点となる点を抽出する (S 4 7 0)。これを模式的に示すのが図 3 4 である。図 3 4 において黒四角形で示される点が S 4 7 0 で抽出した点である。

【 0 1 2 2 】

そして S 4 7 0 で抽出した各点の E 点に、外部の石が 2 以上含まれ、判定対象の石が 1 つも含まれていない点があるかを判定する。そしてある場合には、その点に判定対象となる相手側の石 (白石) を配置し、外部としてみなす (S 4 8 0)。これを模式的に示すのが図 3 5 である。

【 0 1 2 3 】

そして S 4 8 0 において一つ以上の石を配置した場合には、再度、S 4 7 0 乃至 S 4 8 0 の処理を実行する (S 4 9 0)。もし石を配置しなかった場合には、その時点で第 2 外

10

20

30

40

50

部処理部 2 1 7 の処理を終了する (S 4 9 0)。図 3 5 の場合、 S 4 8 0 の条件を充足しないので、新たに石は配置されない。

【 0 1 2 4 】

以上のようにして第 2 外部処理部 2 1 7 における処理が終了後、死活判定実行部 2 1 8 が S 4 9 0 の処理終了後の盤面の情報に対して、総あたり処理を実行する。上述の例の場合、図 3 6 の各点に対して総あたりの死活判定処理を死活判定実行部 2 1 8 が実行することとなる (S 5 0 0)。

【 0 1 2 5 】

そしてこの盤面の情報で死活判定処理の結果、生き (勝ち) となる点が 1 点以上存在している場合には (S 5 1 0)、死活判定処理として生き (勝ち) として最終結果を判定する (S 5 2 0)。

10

【 0 1 2 6 】

一方、 S 5 0 0 の死活判定処理の結果、生き (勝ち) となる点が 1 点も存在しない場合には (S 5 1 0)、盤面の情報を初期状態 (図 2 7 の状態) に戻し、死活判定実行部 2 1 8 が通常どおりの死活判定処理を実行し、その結果を最終結果として判定する。

【 0 1 2 7 】

以上のようにして死活判定処理部 2 1 b で死活判定の処理結果を判定すると、出力処理部 2 2 は、その処理結果を対局者端末 3 に送信する。

【 0 1 2 8 】

本実施例のような処理を実行したあとで死活判定実行部 2 1 8 が死活判定処理の総あたり処理を実行することで、判定領域内の総あたりの対象となる点を減らすことが出来る。従って、総あたり処理にかかる時間を減らすことが出来るので、処理結果を早く算出することが出来る。

20

【 0 1 2 9 】

上述では編集後の盤面の情報が図 2 7 の状態に対する処理を説明したが、次に編集後の盤面の情報が図 3 7 であったとする。この場合の実施例 2 の処理プロセスを以下に説明する。

【 0 1 3 0 】

例えば図 3 7 の画面における「判定処理」を押下することで、編集後の盤面の情報 (棋譜の情報と判定対象の石の情報など) と手番の情報と、死活判定処理を開始することの要求とが対局者端末 3 からサーバ 2 に送信される。サーバ 2 の入力受付部 2 0 で編集後の盤面の情報と手番の情報と死活判定処理を開始することの要求を受け付けると、死活判定処理部 2 1 b は、入力を受け付けた編集後の盤面の情報に基づいて、死活判定処理を開始する。

30

【 0 1 3 1 】

まず死活判定処理部 2 1 b の第 1 外部処理部 2 1 6 は、総あたり処理を開始するにあたり、盤面の棋譜の情報を抽出し、判定対象の石を含む閉領域を構成する石の群を抽出する (S 4 0 0)。

【 0 1 3 2 】

そうすると、図 3 7 の場合、閉領域を構成する石の群が抽出できない。そのため、第 1 外部処理部 2 1 6 はその処理を中止し、第 2 外部処理部 2 1 7 における処理を実行する (S 4 1 0)。

40

【 0 1 3 3 】

そして第 2 外部処理部 2 1 7 は、外部の E 空点であって、判定領域となる点を抽出する (S 4 7 0)。これを模式的に示すのが図 3 8 である。図 3 8 において黒四角形で示される点が S 4 7 0 で抽出した点である。

【 0 1 3 4 】

そして S 4 7 0 で抽出した各点の E 点に、外部の石が 2 以上含まれ、判定対象の石が 1 つも含まれていない点があるかを判定する。そしてある場合には、その点に判定対象となる相手側の石 (白石) を配置し、外部としてみなす (S 4 8 0)。これを模式的に示すの

50

が図 39 である。そして S 480 の処理を実行すると、新たに相手側の石が図 40 のように配置される。なお図 40 では新たに配置される相手側の石を星で示している。なおここでは判定したすべての点に相手側の石を配置したが、すべてではなく、一部の点だけでも良い。

【 0 1 3 5 】

S 480 で新たに相手側の石を配置したので (S 490)、第 2 外部処理部 217 は、外部の E 空点であって、判定領域となる点を抽出する (S 470) (なお本実施例では新たに相手側の石を配置した場合に反復処理を実行するようにしたが、反復処理を実行せずとも良い)。これを模式的に示すのが図 41 である。図 41 において黒四角形で示される点が S 470 で抽出した点である。

10

【 0 1 3 6 】

そして S 470 で抽出した各点の E 点に、外部の石が 2 以上含まれ、判定対象の石が 1 つも含まれていない点があるかを判定する。そしてある場合には、その点に判定対象となる相手側の石 (白石) を配置し、外部としてみなす (S 480)。これを模式的に示すのが図 42 である。そして S 480 の処理を実行すると、新たに相手側の石が図 43 のように配置される。なお図 43 では新たに配置される相手側の石を星で示している。

【 0 1 3 7 】

S 480 で新たに相手側の石を配置したので (S 490)、第 2 外部処理部 217 は、外部の E 空点であって、判定領域となる点を抽出する (S 470)。これを模式的に示すのが図 44 である。図 44 において黒四角形で示される点が S 470 で抽出した点である。

20

【 0 1 3 8 】

そして S 470 で抽出した各点の E 点に、外部の石が 2 以上含まれ、判定対象の石が 1 つも含まれていない点があるかを判定する。そしてある場合には、その点に判定対象となる相手側の石 (白石) を配置し、外部としてみなす (S 480)。これを模式的に示すのが図 45 である。

【 0 1 3 9 】

図 45 の状態では新たに石を配置することがない。そのため、第 2 外部処理部 217 の処理を終了する (S 490)。

【 0 1 4 0 】

30

以上のようにして第 2 外部処理部 217 における処理が終了後、死活判定実行部 218 が S 490 の処理終了後の盤面の情報に対して、総あたり処理を実行する。上述の例の場合、図 46 の各点に対して総あたりの死活判定処理を死活判定実行部 218 が実行することとなる (S 500)。

【 0 1 4 1 】

そしてこの盤面の情報で死活判定処理の結果、生き (勝ち) となる点が 1 点以上存在している場合には (S 510)、死活判定処理として生き (勝ち) として最終結果を判定する (S 520)。

【 0 1 4 2 】

一方、S 500 の死活判定処理の結果、生き (勝ち) となる点が 1 点も存在しない場合には (S 510)、盤面の情報を初期状態 (図 27 の状態) に戻し、死活判定実行部 218 が通常どおりの死活判定処理を実行し (S 530)、その結果を最終結果として判定する。

40

【 0 1 4 3 】

以上のようにして死活判定処理部 21b で死活判定の処理結果を判定すると、出力処理部 22 は、その処理結果を対局者端末 3 に送信する。

【 0 1 4 4 】

次に上述の各処理のより詳細な説明を図 47 乃至図 53 を用いて説明する。

【 0 1 4 5 】

まず死活判定処理部 21b は、盤面の情報を読み込み、Board [x] [y] に空点

50

であるか、黒石が置かれているか、白石が置かれているかを記憶していく。また判定領域 $Area[x][y]$ にそれぞれ判定領域であるか否かを記憶していく。また上記のほか、判定対象の石 $H[x][y]$ 、判定領域 $Area[x][y]$ 、手番 tn など適宜、設定される。そして、死活判定処理部 21b における第 1 外部処理部 216 がそこにおける処理（処理 A）を実行する（S600）。

【0146】

すなわち、まずすべての点 (x, y) について $R[x][y] = Area[x][y]$ 、 $Q[x][y] = Area[x][y]$ を代入する（S6000）。そして初期値として変数 f に $f = 0$ を代入する（S6010）。

【0147】

次に第 1 外部処理部 216 は、 $Area[x][y] = 1$ 、 $Board[x][y] = 0$ となる点 (x, y) について、S6020 乃至 S6080 の処理を反復する。

【0148】

まず $Q[x][y] = 1$ であるかを判定し（判定領域であるかを判定し）（S6020）、そうでなければ、次の変数 x, y の処理に移る。一方、 $Q[x][y] = 1$ であるならば（判定領域であるならば）、当該点 (x, y) を含む閉領域を抽出し、それを配列 $C[x][y]$ に代入する（S6030）。そして $Q[x][y] = 0$ を代入する（S6040）。

【0149】

$C[x][y] = 1$ でないならば（S6050）、次の変数 x, y の処理に移る。一方、 $C[x][y] = 1$ であるならば、 $C[a][b] > 0$ である点すべてについて（1 a, b 19）、 $Q[a][b] = 0$ を代入する（S6060）。そして後述する処理 A2 を実行する（S6070）。なおこの処理は、図 24 における S420 に相当する処理である。

【0150】

そして第 1 外部処理部 216 は、処理 A2 を実行する。すなわち、配列 $C[x][y]$ の E 点を抽出し、それを配列 $E[x][y]$ に代入する（S6073）。また配列 $C[x][y]$ の O 点で 1 線の点を抽出し、それを配列 $O[x][y]$ に代入する（S6075）。その後、配列 $C[c][d] = 1$ 、配列 $E[c][d] = 1$ 、または配列 $O[c][d] = 1$ のときには、 $R[c][d] = 0$ を代入する（1 c, d 19）（S6077）。

【0151】

以上のようにして第 1 外部処理部 216 が処理 A2 を実行すると、変数 f について、 $f = 1$ を代入する（S6080）。以上の処理を第 1 外部処理部 216 は、 $Area[x][y] = 1$ 、 $Board[x][y] = 0$ となるすべての点 (x, y) について反復処理を実行する。

【0152】

以上のような処理を実行することで、図 24 における S400 乃至 S440 に相当する処理を実行することが出来る。

【0153】

上記反復処理が終了後、変数 f について $f = 1$ であるかを判定し（S6090）、 $f = 1$ であれば、処理 A3 を実行する。もし $f = 1$ でなければ処理 A を中止し、次の処理、すなわち第 2 外部処理部 217 での処理 B を実行する（S610）。

【0154】

上述の S6090 において、 $f = 1$ であるならば、続いて第 1 外部処理部 216 は、処理 A3 を実行する。

【0155】

すなわち、まず配列 $H[x][y] = 1$ のすべての点 (x, y) について、 $R[x][y] = 0$ を代入する（S6100）。次に配列 $H[x][y] = 1$ となる E 点を配列 $E2$ に代入する（S6110）。

10

20

30

40

50

【0156】

$E2[x][y] = 1$ となるすべての点 (x, y) について、配列 $R[x][y] = 0$ を代入する(S6120)。そしてすべての点 (x, y) で配列 $W[x][y] = 0$ を代入する(S6130)。その後、 $Board[x][y] = 0$ 、 $R[x][y] = 1$ となるすべての点 (x, y) について、 $Board[x][y] = tn^3$ 、 $W[x][y] = 1$ 、 $Area[x][y] = 0$ を代入する(S6140)。

【0157】

以上のような処理を実行することで、図24におけるS450に相当する処理を実行することが出来る。そしてS6140の処理終了後、第1外部処理部216は、処理A4を実行する。

10

【0158】

まず第1外部処理部216は、 $Board[x][y] = tn^3$ 、 $Area[x][y] = 0$ 、 $W[x][y] = 0$ となるすべての点 (x, y) をグループ化し、配列 $G2[x][y]$ に代入する(S6150)。そして $W[x][y] = 1$ となるすべての点 (x, y) について、S6160乃至S6170の反復処理を実行する。

【0159】

まず配列 $G2[x][y] = 0$ であるかを判定し(S6160)、もし0でないならば、次の点 (x, y) の処理に移る。そして0であるならば、 $Area[x][y] = 1$ 、 $Board[x][y] = 0$ を代入する(S6170)。そして次の点 (x, y) の処理に移る。

20

【0160】

以上の反復処理を $W[x][y] = 1$ となるすべての点 (x, y) について実行することで、処理A4が終了する。以上のような処理を実行することで、図24におけるS460に相当する処理を実行することが出来る。

【0161】

以上の処理が終了した場合、あるいはS6090で $f = 1$ でない場合、第2外部処理部217での処理Bが実行される(S610)。

【0162】

まず第2外部処理部217は、 $Board[x][y] = tn^3$ 、 $Area[x][y] = 0$ となる点 (x, y) のE空点を配列 $E1[x][y]$ に代入する(S6200)。そして次に $Area[x][y] = 0$ となる点 (x, y) について、配列 $E1[x][y] = 0$ を代入する(S6210)。以上によって、図24におけるS470に相当する処理を実行することが出来る。

30

【0163】

そして次に配列 $E1[x][y] = 1$ となるすべての点 (x, y) について、S6220乃至S6300の反復処理を実行する。

【0164】

まず点 (x, y) のE点を配列 $E2[x][y]$ に代入し(S6220)、変数 $C = 0$ を代入する(S6230)。そして配列 $E2[x'][y'] = 1$ となる点 (x', y') について、S6240乃至S6270の反復処理を実行する。

40

【0165】

まず配列 $H[x'][y'] = 1$ であるかを判定し(S6240)、そうであるならば、変数 C から4を減算して C を更新する(S6250)。すなわち $C = C - 4$ を実行する。そして、 $H[x'][y'] = 1$ でない場合、あるいはS6250を実行した場合、 $Area[x'][y'] = 0$ であるかを判定する(S6260)。そうであるならば変数 C に1を加算して C を更新する(S6270)。すなわち $C = C + 1$ を実行する。もしそうでない場合、あるいはS6270を実行した場合、配列 $E2[x'][y'] = 1$ となる次の点 (x', y') について、同様の処理を実行する。

【0166】

配列 $E2[x'][y'] = 1$ となるすべての点 (x', y') について処理を実行し

50

た後、C 2であるかを判定する(S 6 2 8 0)。もしC 2であるならば、点(x, y)が1線であるかを判定する(S 6 2 9 0)。一方、C 2ではないならば、配列E 1 [x] [y] = 0を代入する(S 6 3 0 0)。またS 6 2 9 0において、点(x, y)が1線であるならば、配列E 1 [x] [y] = 0を代入する(S 6 3 0 0)。

【0 1 6 7】

S 6 2 9 0において点(x, y)が1線ではない場合、またはS 6 3 0 0において配列E 1 [x] [y] = 0を代入した場合、配列E 1 [x] [y] = 1となる次の点(x, y)について、上述と同様の処理を実行する。

【0 1 6 8】

以上のように配列E 1 [x] [y] = 1となるすべての点(x, y)についてS 6 2 2 0乃至S 6 3 0 0の処理が終了後、配列E 1 [x] [y] = 1となる点があるかを判定する(S 6 3 1 0)。もし配列E 1 [x] [y] = 1となる点がない場合には、第2外部処理部2 1 7における処理を終了し、現在の盤面の状態における死活判定実行部2 1 8での総あたりでの死活判定処理を実行させる(S 6 2 0)。

【0 1 6 9】

一方、S 6 3 1 0において配列E 1 [x] [y] = 1となる点がある場合には、配列E 1 [x] [y] = 1となる点について、Board [x] [y] = t n ^ 3、Area [x] [y] = 0を代入する(S 6 3 2 0)。そして再度、処理Bに戻り、第2外部処理部2 1 7が上述の処理を実行する。すなわちS 6 2 0 0以降の処理を実行する。

【0 1 7 0】

以上の処理を実行することで、図2 4におけるS 4 8 0乃至S 4 9 0に相当する処理を実行することが出来る。

【0 1 7 1】

そして以上のようにして処理A、処理Bを実行すると、当初の状態から盤面の状態が変形され、判定領域となる点が狭められる。そしてこの狭められた点に対して、死活判定実行部2 1 8が総あたりでの死活判定処理を従来と同様に実行する(S 6 2 0)。

【0 1 7 2】

そしてこの死活判定処理の結果、生き(勝ち)となる点が1点以上存在している場合には(S 6 3 0)、死活判定処理として生き(勝ち)として最終結果を判定する(S 6 4 0)。

【0 1 7 3】

一方、S 6 2 0の死活判定処理の結果、生き(勝ち)となる点が1点も存在しない場合には(S 6 3 0)、盤面の情報を初期状態に戻し、死活判定実行部2 1 8が通常どおりの死活判定処理を実行し(S 6 5 0)、その結果を最終結果として判定する。

【0 1 7 4】

以上のようにして死活判定処理部2 1 bで死活判定の処理結果を判定すると、出力処理部2 2は、その処理結果を対局者端末3に送信する。

【0 1 7 5】

本実施例のように、単純な総あたりによる死活判定処理を実行する前に、外部を基準として、外部の石の色と同色の石を所定の条件で判定領域内に新たに配置することで、死活判定処理の際の総あたりの処理対象を減らすことが出来る。そして、外部を狭めて死活判定をし、それでも活着している場合には、もとの盤面の状態でも生きと判定する処理を実行することで、判定対象となる石の死活判定処理を高速に実現することが出来る。

【実施例3】

【0 1 7 6】

次に、死活判定処理部2 1 cにおける高速処理の第3の実施例を以下に説明する。本実施例における死活判定処理部2 1 cの処理機能の一例の概念図を図5 4に示す。本実施例の死活判定処理部2 1 cでは、総あたりの死活判定処理を行うにあたり、判定領域の盤面を所定の方法で分割し、それぞれの眼形に変形した上で、死活判定処理を実行する場合を示す。つまり、判定領域の盤面を所定の方法で分割し、分割した領域のうち一の領域につ

10

20

30

40

50

いて、もっとも不利な眼形の形状に変形した上で、判定領域全体の死活判定処理を実行する場合である。

【 0 1 7 7 】

本実施例の死活判定処理部 2 1 c には、領域分割処理部 2 1 9 と 0 眼変形処理部 2 2 0 と後手 1 眼変形処理部 2 2 2 と 1 眼変形処理部 2 2 2 と死活判定実行部 2 2 3 とを備える。

【 0 1 7 8 】

領域分割処理部 2 1 9 は、判定領域において、外部と隣接している判定対象の石から、同色で隣接している石のつながりを再帰的に判定し、もう一方の端が別の外部と隣接しているかを判定する。このような石のつながりを判定することで、判定領域における盤面を分割する。

10

【 0 1 7 9 】

0 眼変形処理部 2 2 0 は、領域分割処理部 2 1 9 で判定した領域のうちの一つの領域について、0 眼となるように変形を行う。具体的には、当該領域に対して最も判定対象側に不利となる 0 眼となるように変形させる。例えばすべての点を相手の石を配置することで、変形を行う。例えば図 5 6 に示すように、分割線に隣接する点すべてに外部と同じ石を配置する。図 5 6 では星で示した石が新たに配置される石である。

【 0 1 8 0 】

後手 1 眼変形処理部 2 2 1 は、領域分割処理部 2 1 9 で判定した領域のうちの一つの領域について、後手 1 眼となるように変形を行う。具体的には、当該領域に対して最も判定対象側に不利となる後手 1 眼となるように変形させる。例えば当該領域において、分割線に隣接させた相手の 2 つの石が取り得る状態となるような後手 1 眼となるように変形を行う。例えば図 5 7 に示すように、黒から打つと 1 眼、白から打つと 0 眼となるように配置する。図 5 7 では星で示した石が新たに配置される石である。

20

【 0 1 8 1 】

1 眼変形処理部 2 2 2 は、領域分割処理部 2 1 9 で判定した領域のうちの一つの領域について、1 眼となるように変形を行う。具体的には当該領域に対して最も判定対象側に不利となる 1 眼となるように変形させる。例えば当該領域において、分割線に隣接させた相手の 2 つの石が取り得る状態となる 1 眼となるように変形を行う。例えば図 5 8 に示すように、黒から打っても白から打っても黒 1 眼となるように配置する。図 5 8 では星で示した石が新たに配置される石である。

30

【 0 1 8 2 】

なお 0 眼、後手 1 眼、1 眼への変形はどのような変形を行うことも出来、如何なる処理方法を採用しても良い。

【 0 1 8 3 】

死活判定実行部 2 2 3 は、分割した領域に対して 0 眼変形処理部 2 2 0、後手 1 眼変形処理部 2 2 1、1 眼変形処理部 2 2 2 のいずれかにおいて変形を行ったあと、その判定領域の全体に対して、死活判定処理を行う。また本来の判定領域に対して、死活判定処理を行う。この死活判定処理は従来と同様の死活判定処理プログラムを用いることが出来る。なお以下の実施例では、総あたりによる死活判定処理を行う場合を示すが、それ以外の従来の死活判定処理プログラムによる方法を用いることも可能である。また総あたりでの死活判定処理とは、従来の死活判定処理プログラムにおいて用いられている死活判定処理であっても良く、必ずしも判定領域内のすべての空点に対する総あたり処理を行わなくても良い。例えば判定領域内の空点のうち、一部の空点については処理を省略するなどの方法であっても良い。例えば、価値が全く同じ着手が複数あった場合、一つの着手のみを実行する、といったように処理を省略する方法などがある。すなわち、死活判定実行部 2 2 3 は、任意の方法により、判定領域内にある判定対象の石または目標石に対しての死活判定処理を実行できる方法であれば如何なる方法であっても良い。

40

【 0 1 8 4 】

次に本実施例における死活判定処理の処理プロセスの一例を図 5 5 のフローチャートを

50

用いて説明する。

【 0 1 8 5 】

対局者が利用する対局者端末 3 同士でネットワークを介して囲碁の対局が行われている。この際に囲碁の対局は対局システムを実現するサーバ 2 上でその処理が行われている。そしてその対局中、あるいは終局後、対局者のいずれかがその表示される画面から所定のボタン等を押下することで、死活判定処理を行うことの要求が対局者端末 3 から、死活判定処理を行うサーバ 2 に送信される。対局画面の一例を図 5 9 に示す。

【 0 1 8 6 】

対局画面が図 5 9 の場合、その画面のうち、「死活ナビ」など、死活判定処理を行うことを要求するボタンを押下することで、死活判定処理を行うことの要求が対局者端末 3 からサーバ 2 に送信される。

【 0 1 8 7 】

サーバ 2 の入力受付部 2 0 でその要求を受け付けると、死活判定処理部 2 1 c は、対局システムの記憶装置 1 1 から当該盤面の情報（たとえば棋譜の情報）を取得する。そして盤面の情報が含まれた死活判定処理の画面を出力処理部 2 2 が対局者端末 3 に送信する。この際の画面の一例が図 6 0 である。なお実際に対局している盤面の情報のほか、すでに対局し、所定の記憶装置 1 1 に保存している棋譜の情報、あるいは観戦している対局の棋譜の情報を死活判定処理部 2 1 c が対局システムの記憶装置 1 1 から取得し、その棋譜の情報を図 6 0 の画面に含めて対局者端末 3 に送信しても良い。

【 0 1 8 8 】

死活判定を行う対局者は、図 6 0 に示す死活判定処理を行う画面から盤面の編集を行う。例えば表示されている石を指定することで、石を取り除いたり、あるいは石を置く。またマウスなどの入力装置 1 3 をドラッグするなどにより、判定領域を指定する。このように死活判定処理を所望する対局者は、適宜、盤面の情報を編集する。この際の編集される石の情報、手番の情報などは、対局者端末 3 からサーバ 2 に適宜、送られる。

【 0 1 8 9 】

上述のように対局者が判定領域を指定することにより、その判定領域内にある結論を出すべき石を判定対象の石として設定される。

【 0 1 9 0 】

編集後の死活判定処理を行う画面の一例を図 6 1 に示す。なお図 6 1 では判定対象の石が黒の 9 子、手番は黒であるとする。従って、この場合、次の順番は黒石であり、さらに黒石から見て黒の 9 子の死活判定を行うことを意味している。

【 0 1 9 1 】

例えば図 6 1 の画面における「判定処理」を押下することで、編集後の盤面の情報（棋譜の情報と判定対象の石の情報など）と手番の情報と、死活判定処理を開始することの要求とが対局者端末 3 からサーバ 2 に送信される。サーバ 2 の入力受付部 2 0 で編集後の盤面の情報と手番の情報と死活判定処理を開始することの要求を受け付けると、死活判定処理部 2 1 c は、入力を受け付けた編集後の盤面の情報に基づいて、死活判定処理を開始する。

【 0 1 9 2 】

まず死活判定処理部 2 1 c の領域分割処理部 2 1 9 は、総あたり処理を開始するにあたり、盤面の棋譜の情報を抽出し、その判定領域を分割することで、分割可能であるかを判定する（S700）。すなわち、外部と隣接している判定対象の石から、同色で隣接している石のつながりを再帰的に判定し、他方の端が別の外部と隣接しているかを判定する。そしてこのような石のつながりを検索し、石のつながりがあった場合には、判定領域が分割可能であると判定する。なお以下の実施例では本来の判定領域を 2 つの領域に分割する場合を説明するが、3 以上の領域に分割できる場合、そのうちの 1 つと残りの全部の領域の判定を行う。そして残りの領域について判定する際に、上述の分割可能かの判定を再帰的に行うことで、3 以上の領域に分割できる場合であっても、以下の実施例と同様に処理が行える。

10

20

30

40

50

【 0 1 9 3 】

上述のように S 7 0 0 で、領域分割処理部 2 1 9 が、本来の判定領域が分割可能ではないと判定した場合には (S 7 1 0)、そのまま処理を中止し、死活判定実行部 2 2 3 が、本来の盤面の状態で通常どおりの総あたりの死活判定処理を実行する (S 8 4 0)。

【 0 1 9 4 】

一方、S 7 0 0 で、領域分割処理部 2 1 9 が、本来の判定領域が分割可能であると判定した場合には (S 7 1 0)、その判定した領域について 2 つに分割する。この状態を模式的に示すのが図 6 2 である。図 6 2 では黒星で示される石が分割線となる石である。

【 0 1 9 5 】

そして分割線を境に 2 つの領域に判定領域を分割する。ここでは分割線より上方を領域 R 1、下方を領域 R 2 とする。この状態を示すのが図 6 3 である。なお上下に分割するほか、左右に分割するなど、如何なる分割であっても良い。

【 0 1 9 6 】

このように領域分割処理部 2 1 9 で 2 つの領域に分割後、いずれか一方の領域 (ここでは領域 R 1 とする) について、まず 0 眼変形処理部 2 2 0 が 0 眼となるように変形を行う (S 7 2 0)。例えば、領域 R 1 に含まれる石を取り除き、そこに 0 眼となるように、分割線に隣接する点すべてに外部と同じ石 (ここでは白石) を配置する。この状態を模式的に示すのが図 6 4 である。ここで白星が新たに配置された石である。

【 0 1 9 7 】

そして S 7 2 0 で変形処理後、変形処理後の盤面の状態に対して、死活判定実行部 2 2 3 が総あたりによる死活判定処理を実行する (S 7 3 0)。死活判定実行部 2 2 3 による死活判定処理の結果、「勝ち」の場合には (S 7 4 0)、勝ちと判定できるので (S 7 5 0)、そのまま処理を終了する。図 6 4 の場合、活かないので、S 7 4 0 の処理に移る。

【 0 1 9 8 】

すなわち、S 7 3 0 における死活判定実行部 2 2 3 による死活判定処理の結果、「負け」と判定した場合には (S 7 4 0)、次に、後手 1 眼変形処理部 2 2 1 が、領域 R 1 に対して、後手 1 眼となるように変形を行う (S 7 6 0)。例えば、領域 R 1 に含まれる石を取り除き、そこに後手 1 眼となるように、例えば黒から打つと 1 眼、白から打つと 0 眼となるように、配置する。この状態を模式的に示すのが図 6 5 である。ここで白星、黒星が新たに配置された石である。

【 0 1 9 9 】

そして S 7 6 0 で変形処理終了後、変形処理後の盤面の状態に対して、死活判定実行部 2 2 3 が総あたりによる死活判定処理を実行する (S 7 7 0)。死活判定実行部 2 2 3 による死活判定処理の結果、「勝ち」の場合には (S 7 8 0)、次に、盤面をもとの状態に戻し、分割した領域のうち S 7 6 0 で処理した領域とは別の領域、すなわち領域 R 2 について、1 眼変形処理部 2 2 2 が 1 眼となるように変形を行う (S 7 9 0)。例えば、領域 R 2 に含まれる石を取り除き、そこに 1 眼となるように、例えば黒から打つても白から打つても黒 1 眼となるように、配置する。この状態を模式的に示すのが図 6 6 である。ここで白星、黒星が新たに配置された石である。

【 0 2 0 0 】

そして S 7 9 0 で変形処理終了後、変形処理後の盤面の状態に対して、死活判定実行部 2 2 3 が総あたりによる死活判定処理を実行する (S 8 0 0)。死活判定実行部 2 2 3 による死活判定処理の結果、「勝ち」の場合には (S 8 7 0)、勝ちと判定できるので (S 8 8 0)、そのまま処理を終了する。図 6 7 の場合、活きるので、処理を終了する。

【 0 2 0 1 】

一方、S 8 7 0 において死活判定実行部 2 2 3 で「負け」と判定した場合には (S 8 7 0)、そのまま処理を中止し、死活判定実行部 2 2 3 が、本来の盤面の状態で通常どおりの総あたりの死活判定処理を実行する (S 8 4 0)。

【 0 2 0 2 】

また、S 7 8 0 において死活判定実行部 2 2 3 で活きないと判定した場合には (S 7 8 0)、次に、1 眼変形処理部 2 2 2 が、領域 R 1 に対して、1 眼となるように変形を行う (S 8 1 0)。例えば、領域 R 1 に含まれる石を取り除き、そこに 1 眼となるように、例えば黒から打っても白から打っても黒 1 眼となるように、配置する。

【 0 2 0 3 】

そして S 8 1 0 で変形処理終了後、変形処理後の盤面の状態に対して、死活判定実行部 2 2 3 が総あたりによる死活判定処理を実行する (S 8 2 0)。死活判定実行部 2 2 3 による死活判定処理の結果、「負け」の場合には (S 8 3 0)、盤面をもとの状態に戻し、分割した領域のうち S 8 1 0 で処理した領域とは別の領域、すなわち領域 R 2 について、0 願変形処理部 2 2 0 が 0 眼となるように変形を行う (S 8 3 3)。そして S 8 3 3 で変形処理終了後、変形処理終了後の盤面の状態に対して、死活判定実行部 2 2 3 が総あたりによる死活判定処理を実行する (S 8 3 5)。死活判定実行部 2 2 3 による死活判定処理の結果、「勝ち」の場合には (S 8 3 7)、勝ちと判定できるので (S 8 8 0)、そのまま処理を終了する。一方、S 8 3 5 の死活判定実行部 2 2 3 で「負け」と判定した場合には (S 8 3 7)、そのまま処理を中止し、死活判定実行部 2 2 3 が、本来の盤面の状態で通常どおりの総あたりの死活判定処理を実行する (S 8 4 0)。

10

【 0 2 0 4 】

一方、S 8 2 0 において死活判定実行部 2 2 3 で「勝ち」と判定した場合には (S 8 3 0)、次に、盤面をもとの状態に戻し、分割した領域のうち S 8 1 0 で処理した領域とは別の領域、すなわち領域 R 2 について、後手 1 眼変形処理部 2 2 1 が後手 1 眼となるように変形を行う (S 8 5 0)。例えば、領域 R 2 に含まれる石を取り除き、そこに後手 1 眼となるように、例えば黒から打つと 1 眼、白から打つと 0 眼となるように、配置する。

20

【 0 2 0 5 】

そして S 8 5 0 で変形処理終了後、変形処理後の盤面の状態に対して、死活判定実行部 2 2 3 が総あたりによる死活判定処理を実行する (S 8 6 0)。死活判定実行部 2 2 3 による死活判定処理の結果、「勝ち」の場合には (S 8 7 0)、勝ちと判定できるので (S 8 8 0)、そのまま処理を終了する。

【 0 2 0 6 】

一方、S 8 6 0 において死活判定実行部 2 2 3 で「負け」と判定した場合には (S 8 7 0)、そのまま処理を中止し、死活判定実行部 2 2 3 が、本来の盤面の状態で通常どおりの総あたりの死活判定処理を実行する (S 8 4 0)。

30

【 0 2 0 7 】

以上のように、総あたりの死活判定処理を行うにあたり、判定領域の盤面を所定の方法で分割し、それぞれの眼形に変形した上で、全体の判定領域の死活判定処理を実行することで、本来の判定領域に対して単純に総あたりの死活判定処理を実行するよりも高速に死活判定処理の結果を知ることが出来る。これは、判定対象の石に 2 眼があれば生きることに着目したものである。すなわち判定領域を 2 つに分割し、そのうちの 1 つの領域について 0 眼とした場合、他方の領域が生きていればそこが 2 眼となっているので、全体として判定対象の石が活けると判定できる。また 1 つの領域について後手 1 眼とした場合、他方の領域が生きていればそこに 1 眼があることとなるので、全体として判定対象の石が活けると判定できる。さらに 1 つの領域について 1 眼とした場合、他方の領域が生きていればそこに後手 1 眼があることとなるので、全体として判定対象の石が活けると判定できる。

40

【 0 2 0 8 】

このように、活きる石は 2 眼を持つことが多いことに着目して、判定領域の盤面を分割し、分割した領域を 0 眼、後手 1 眼、1 眼のいずれかに変形させ、変形させた盤面の状態に対して、総あたりで死活判定処理で活きるかを判定することで、本来の盤面の死活判定処理を行うよりも高速に死活判定処理の結果を得ることが可能となる。

【 0 2 0 9 】

以上のようにして死活判定処理部 2 1 c で死活判定の処理結果を判定すると、出力処理部 2 2 は、その処理結果を対局者端末 3 に送信する。

50

【 0 2 1 0 】

次に上述の各処理のより詳細な説明を図 6 7 乃至図 7 1 を用いて説明する。

【 0 2 1 1 】

まず死活判定処理部 2 1 c は、盤面の情報を読み込み、Board [x] [y] に空点であるか、黒石が置かれているか、白石が置かれているかを記憶していく。また判定領域 Area [x] [y] にそれぞれ判定領域であるか否かを記憶していく。また上記のほか、判定対象の石 H [x] [y]、判定領域 Area [x] [y]、手番 t n など適宜、設定される。そして、死活判定処理部 2 1 c における領域分割処理部 2 1 9 が判定領域を分割し、分割可能かを判定する。すなわち処理 D 1、D 2 を実行させる (S 9 0 0)。

【 0 2 1 2 】

領域分割処理部 2 1 9 は、外部の E 点で、かつ Board [x] [y] = t n となる点について配列 R [x] [y] = 1 を代入する (S 1 0 3 0)。そして、R [x] [y] = 1 となる点 (x , y) について、S 1 0 4 0 乃至 S 1 0 9 0 を反復処理させる。

【 0 2 1 3 】

まず R [x] [y] = 1 となる点 (x , y) に配列 T [x] [y] = 1 を代入し、それ以外の点に T [x] [y] = 0 を代入する (S 1 0 4 0)。そして点 (x , y) の E 点で Board [x '] [y '] = t n、T [x '] [y '] = 0 となる点 (x ' , y ') について、S 1 0 5 0 乃至 S 1 0 9 0 を反復処理させる。

【 0 2 1 4 】

まず T [x '] [y '] = 1 を代入する (S 1 0 5 0)。そして、点 (x ' , y ') の E 点に外部が含まれるかを判定し (S 1 0 6 0)、外部に含まれる場合には後述する処理 D 2 を実行させる。一方、外部に含まれない場合には点 (x ' , y ') の E 点 (a , b) で T [a] [b] = 1 となる点が 1 点であるかを判定する (S 1 0 7 0)。なおここで (a = x ' + 1 , x ' - 1 , b = 0)、または、(b = y ' + 1 , y ' - 1 , a = 0) である。そして 1 点ではない場合には T [x '] [y '] = 0 とする (S 1 1 0 0)。また 1 点である場合には点 (x ' , y ') の E 点 (c , d) で T [c] [d] = 0、Board [c] [d] = t n となる点があるかを判定する (S 1 0 8 0)。なおここで (c = x ' + 1 , x ' - 1 , d = 0)、または、(d = y ' + 1 , y ' - 1 , c = 0) である。そして S 1 0 8 0 の条件を満たす点がない場合には、T [x '] [y '] = 0 とする (S 1 1 0 0)。

【 0 2 1 5 】

一方、点 (x ' , y ') の E 点 (c , d) で T [c] [d] = 0、Board [c] [d] = t n となる点がある場合には (S 1 0 8 0)、点 (x , y) に点 (x ' , y ') を代入する (S 1 0 9 0)。以上の処理を、点 (x , y) の E 点で Board [x '] [y '] = t n、T [x '] [y '] = 0 となる点 (x ' , y ') のすべてについて処理を行う。すべての点 (x ' , y ') について処理実行後、点 (x , y) の次の点について処理を実行させる。

【 0 2 1 6 】

以上の処理を、R [x] [y] = 1 となる点 (x , y) のすべてについて処理を行う。もし処理が終了した場合には、分割できないと判定できるので (S 1 1 1 0)、領域分割処理部 2 1 9 は以降の処理を中止し、死活判定実行部 2 2 3 における通常どおりの総あたりによる死活判定処理を実行させる (S 9 1 0、S 1 0 2 0)。

【 0 2 1 7 】

上述の処理の S 1 0 6 0 において、点 (x ' , y ') の E 点に外部が含まれる場合には処理 D 2 が実行されるので、それを説明する。まず配列 T [x] [y] をグループ化し、配列 G 1 [x] [y] に代入する (S 1 1 2 0)。そして配列 H 1 [x] [y] に配列 H [x] [y] を代入する (S 1 1 3 0)。

【 0 2 1 8 】

次に配列 G 1 [x] [y] = 1 となる点 (x , y) について、H 1 [x] [y] = 0 を代入する (S 1 1 4 0)。そして H 1 [x] [y] = 1 となる点 (x , y) を取り、それ

10

20

30

40

50

を配列 $H1[x][y]$ でグループ化し、さらにそれを配列 $H2[x][y]$ に代入する ($S1150$)。そして $H2[x][y] = 1$ となる点 (x, y) について、 $H1[x][y] = 0$ を代入する ($S1160$)。

【0219】

次に、 $H1[x][y] = 1$ となる点 (x, y) が存在するかを判定し ($S1170$)、存在しない場合には処理 $D1$ に戻り、 $T[x'][y'] = 0$ とする ($S1100$)。一方、 $H1[x][y] = 1$ となる点 (x, y) が存在する場合には、判定領域は分割可能であると判定し、各領域を $R1$ 、 $R2$ として分割線 $G1$ を配列 $D[x][y]$ に代入する ($S1180$)。また領域 $R1$ の各点、領域 $R2$ の各点について、それぞれ配列 $R1[x][y]$ に配列 $H1[x][y]$ を代入し、配列 $R2[x][y]$ に配列 $H2[x][y]$ を代入する ($S1180$)。

10

【0220】

以上のような処理を実行することで、図55における $S700$ 乃至 $S710$ に相当する処理を実現できる。

【0221】

そして $S1180$ を終了すると、領域分割処理部219は、判定領域を分割可能であると判定できるので ($S910$)、0眼変形処理部220および死活判定実行部223による処理Aを実行する ($S920$)。

【0222】

まず0眼変形処理部220は、 $R1[x][y] = 1$ となる点 (x, y) について、 $Board[x][y] = tn^3$ 、 $H[x][y] = 0$ とする ($S1190$) (なおここでは領域 $R1$ について0眼への変形処理を実行している)。そして死活判定実行部223において、通常どおりの総あたりの死活判定処理を実行させる ($S1200$)。

20

【0223】

以上のように処理することで、図55における $S720$ 乃至 $S730$ に相当する処理を実行することが出来る。

【0224】

以上のように0眼変形処理部220で0眼への変形処理を行ったあと、死活判定実行部223における死活判定処理を実行した実行結果が勝ちであるならば ($S930$)、そのまま勝ちと判定できるので ($S940$)、処理を終了する。

30

【0225】

一方、死活判定実行部223における死活判定処理を実行した実行結果が負けであるならば ($S930$)、次に後手1眼変形処理部221および死活判定実行部223による処理Bを実行する ($S950$)。

【0226】

まず後手1眼変形処理部221は、領域 $R1$ と、領域 $R1$ の E 点に含まれる外部の点 (x, y) について、 $Board[x][y] = 0$ 、 $H[x][y] = 0$ を代入する ($S1210$)。そして隣接する2点 $P1(x1, y1)$ 、 $P2(x2, y2)$ でそれぞれの E 点が分割線 D の配列 $D[x][y]$ に含まれる点について、 $S1220$ 乃至 $S1260$ 、 $S1280$ 乃至 $S1330$ の処理を反復する。

40

【0227】

まず点 $P1$ 、 $P2$ についてそれぞれ、 $Board[x1][y1] = tn^3$ 、 $Board[x2][y2] = tn^3$ を代入する ($S1220$)。そして点 $P1$ 、 $P2$ の E 点となる点 (x, y) で $Board[x][y] = tn$ を代入する ($S1230$)。そして配列 $D[x][y]$ を再びグループ化し (分割線 D を再びグループ化し)、配列 $D2[x][y]$ に代入する ($S1240$)。そして $D2[x][y] = 1$ となる点の E 空点 (x, y) で、 $Board[x][y] = tn^3$ を代入する ($S1250$)。

【0228】

次に、 $S1230$ で追加した点で、配列 $D2[x][y] = 0$ かつ $S1250$ で追加した点 (x, y) の、 E 点に含まれない点 $P3(x3, y3)$ があるかを判定する ($S12$

50

60)。もしない場合には、次の点の処理に移る。そして2点P1, P2でそれぞれのE点が分割線Dの配列D[x][y]に含まれるすべての点について処理を実行したあと、S1260で条件を充足しない場合には、変形が行えない場合なので、そのまま処理を終了し、死活判定実行部223における死活判定処理を実行する(S1270)。

【0229】

S1260の判定において、条件を充足する点がある場合には、その求めた点P3についてBoard[x3][y3] = tn^3を代入する(S1280)。そして点P1乃至点P3のグループのE空点、O空点となる点(x, y)について、Board[x][y] = tnを代入する(S1290)。そして点P3について、Board[x3][y3] = 0とする(S1300)。

10

【0230】

そして以上までで追加した点のE空点(x, y)について、Board[x][y] = tn^3を代入する(S1310)。ここでは後手1眼変形処理部221で処理Bを実行しているので、通常どおりの総あたりの死活判定処理を実行する(S1330)。

【0231】

以上のように処理することで、図55におけるS760乃至S770に相当する処理を実行することが出来る。

【0232】

以上のように後手1眼変形処理部221で後手1眼への変形処理を行ったあと、死活判定実行部223における死活判定処理を実行した実行結果が勝ちであるならば(S960)、次に1眼変形処理部222および死活判定実行部223で処理C'を実行する(S970)。一方、死活判定処理を実行した実行結果が負けであるならば(S960)、次に1眼変形処理部222および死活判定実行部223で処理Cを実行する(S1000)。

20

【0233】

まず処理C'を実行する場合を説明する。なお図71のフローチャート、すなわち処理B、処理C、処理B'、処理C'で用いるフローチャートは基本的に同じであるが、処理Bと処理B'では処理対象となる領域がR1かR2かで変わる。同様に処理Cと処理C'では処理対象となる領域がR1からR2かで変わる。

【0234】

まず1眼変形処理部222は、領域R2と、領域R2のE点に含まれる外部の点(x, y)について、Board[x][y] = 0、H[x][y] = 0を代入する(S1210)。そして隣接する2点P1(x1, y1), P2(x2, y2)でそれぞれのE点が分割線Dの配列D[x][y]に含まれる点について、S1220乃至S1260、S1280乃至S1330の処理を反復する。

30

【0235】

まず点P1、P2についてそれぞれ、Board[x1][y1] = tn^3、Board[x2][y2] = tn^3を代入する(S1220)。そして点P1、P2のE点となる点(x, y)についてBoard[x][y] = tnを代入する(S1230)。そして配列D[x][y]を再びグループ化し(分割線Dを再びグループ化し)、配列D2[x][y]に代入する(S1240)。そしてD2[x][y] = 1となる点のE空点(x, y)について、Board[x][y] = tn^3を代入する(S1250)。

40

【0236】

次に、S1230で追加した点で、配列D2[x][y] = 0かつS1250で追加した点(x, y)の、E点に含まれない点P3(x3, y3)があるかを判定する(S1260)。もしない場合には、次の点の処理に移る。そして2点P1, P2でそれぞれのE点が分割線Dの配列D[x][y]に含まれるすべての点について処理を実行したあと、S1260で条件を充足しない場合には、変形が行えない場合なので、そのまま処理を終了し、死活判定実行部223における死活判定処理を実行する(S1270)。

【0237】

S1260の判定において、条件を充足する点がある場合には、その求めた点P3につ

50

いて $\text{Board}[x3][y3] = tn^3$ を代入する (S1280)。そして点 P1 乃至点 P3 のグループの E 空点、O 空点となる点 (x, y) について、 $\text{Board}[x][y] = tn$ を代入する (S1290)。そして点 P3 について、 $\text{Board}[x3][y3] = 0$ とする (S1300)。

【0238】

そして以上までで追加した点の E 空点 (x, y) について、 $\text{Board}[x][y] = tn^3$ を代入する (S1310)。ここでは 1 眼変形処理部 222 で処理 C' を実行しているので、次に、S1290 で追加した点の E 点で、配列 $D2[x][y]$ の E 点に含まれない点 (x, y) を一つ求め、 $\text{Board}[x][y] = 0$ を代入する (S1320)。そして死活判定実行部 223 において、通常どおりの総あたりの死活判定処理を実行する (S1330)。

10

【0239】

以上のように処理することで、図 55 における S790 乃至 S800 に相当する処理を実行することが出来る。

【0240】

以上のように 1 眼変形処理部 222 で 1 眼への変形処理を行ったあと、死活判定実行部 223 における死活判定処理を実行した実行結果が勝ちであるならば (S980)、全体として勝ちと判定できるので (S990)、そのまま処理を終了する。一方、死活判定処理を実行した実行結果が負けであるならば (S980)、通常どおり、死活判定実行部 223 における総あたりによる死活判定処理を実行させる (S1020)。

20

【0241】

以上のように処理 B、処理 C、処理 B'、処理 C' を図 71 のフローチャートに基づいて適宜実行することで、死活判定処理による処理結果を得ることが出来る。

【0242】

なお上述の S960 における処理 B において、負けと判定したならば、次に 1 眼変形処理部 222 および死活判定実行部 223 で処理 C を上述と同様に実行する (S1000)。すなわち、図 71 のフローチャートにおいて、処理対象となる領域を R1 として上述の処理 C' と同様の処理を実行する。

【0243】

S1000 における処理 C の結果、すなわち 1 眼変形処理部 222 で 1 眼への変形処理を行った後、死活判定実行部 223 における死活判定処理を実行した結果が勝ちであるならば (S1010)、全体として勝ちと判定できるので (S990)、そのまま処理を終了する。一方、死活判定処理を実行した実行結果が負けであるならば (S1010)、さらに 0 眼変形処理部 220 および死活判定実行部 223 で処理 A' を上述と同様に実行する (S1013)。

30

【0244】

なお処理 A' とは、図 70 に示すフローチャートにおける処理 A と同様の処理であるが、その処理対象となる領域が R1 ではなく、領域 R2 の場合である。

【0245】

そして S1013 における処理 A' の結果、すなわち 0 眼変形処理部 220 で 0 眼への変形処理を行った後、死活判定実行部 223 における死活判定処理を実行した結果が勝ちであるならば (S1015)、全体として勝ちと判定できるので (S990)、そのまま処理を終了する。一方、死活判定処理を実行した実行結果が負けであるならば (S1015)、通常どおり、死活判定実行部 223 における総あたりによる死活判定処理を実行させる (S1020)。

40

【0246】

以上のようにして死活判定処理部 21c で死活判定の処理結果を判定すると、出力処理部 22 は、その処理結果を対局者端末 3 に送信する。

【0247】

本実施例のように、判定領域の盤面を所定の方法で分割し、それぞれの眼形を判定する

50

ことで、活きる石は2眼を持つことが多いという特性を利用した簡易的な判定を行うことが可能となる。

【実施例4】

【0248】

次に、死活判定処理部21dにおける高速処理の第4の実施例を以下に説明する。本実施例における死活判定処理部21dの処理機能の一例の概念図を図72に示す。本実施例の死活判定処理部21dでは、着手を制限し、通常の活きの条件のほかに判定対象の石に隣接する空点が多ければ活きと判定することで、容易に死ぬ石かどうかを判定する方法である。なお、この判定方法を本明細書では緩みシチョウを用いた判定方法という。

【0249】

10

本実施例の死活判定処理部21dには、E空点カウント処理部224と簡易判定処理部225とを備える。

【0250】

E空点カウント処理部224は、判定対象の石のE空点の数をカウントする。

【0251】

簡易判定処理部225は、緩みシチョウを用いることにより、E空点カウント処理部224においてカウントしたE空点の数に基づいて、攻撃側または防御側の着手の優先度を決定し、その優先度に従って攻撃側または防御側の着手を実行し、攻撃側において判定対象の石のE空点の数が4以上の場合に攻撃側の負け、攻撃側において判定対象の石のE空点の数が1の場合に攻撃側の勝ち、と判定する。なお攻撃側または防御側のE空点の数に応じた着手の優先度の決定処理は以下の通りである。

20

【0252】

まず攻撃側について、E空点カウント処理部224においてカウントした、判定対象の石のE空点の数が4点以上の場合には、緩みシチョウで取ることが出来ないので、攻撃側の負けと判定する。

【0253】

攻撃側について、E空点カウント処理部224においてカウントした、判定対象の石のE空点の数が3点の場合、E空点のうち、その点を防御側が着手した時のE空点の数を数え、それが多い順に優先度を決定し、着手を行う。

【0254】

30

攻撃側について、E空点カウント処理部224においてカウントした、判定対象の石のE空点の数が2点の場合、以下の順序で着手を行う。まず第1に、E点となる攻撃側の石のE空点をE空点カウント処理部224でカウントし、1点の場合にはそのE空点に着手する。第2に、E空点のうち、その点を防御側が着手した時のE空点の数をE空点カウント処理部224でカウントし、それが多い順に着手する。第3に、O空点に着手する。第4に、E空点のE空点に着手する。第5に、E空点のO空点に着手する。第6に、O空点のE空点に着手する。

【0255】

攻撃側について、E空点カウント処理部224においてカウントした、判定対象の石のE空点の数が1点の場合は、その点に着手する。この時点で攻撃側の勝ちと判定する。

40

【0256】

次に防御側について、判定対象の石のE空点の数をE空点カウント処理部224でカウントし、それが2点であれば、以下の順序で着手を行う。まず第1に、E点となる攻撃側の石のE空点の数をE空点カウント処理部224でカウントし、1点のときにはそのE空点に着手する。第2に、E点となる攻撃側の石のE空点の数をE空点カウント処理部224でカウントし、2点のときはそのE空点のいずれかに着手する。第3に、E空点のうち、その点を着手した時のE空点の数をE空点カウント処理部224でカウントし、それが多い順に着手する。第4に、O空点に着手する。第5に、E空点のE空点に着手する。第6に、E空点のO空点に着手する。第7に、O空点のE空点に着手する。

【0257】

50

防御側について、判定対象の石のE空点の数をE空点カウント処理部224でカウントし、それが1点であれば、以下の順序で着手を行う。まず第1に、E点となる攻撃側の石のE空点をE空点カウント処理部224でカウントし、それが1点のときにはそのE空点に着手する。第2に、E空点に着手する。

【0258】

以上のような処理を、簡易判定処理部225は、攻撃側および防御側について交互に行い、攻撃側のE空点の数が4点以上、または1点となるまで繰り返し反復する。

【0259】

次に本実施例における死活判定処理の処理プロセスの一例を図73のフローチャートを用いて説明する。

【0260】

対局者が利用する対局者端末3同士でネットワークを介して囲碁の対局が行われている。この際に囲碁の対局は対局システムを実現するサーバ2上でその処理が行われている。そしてその対局中、あるいは終局後、対局者のいずれかがその表示される画面から所定のボタン等を押下することで、死活判定処理を行うことの要求が対局者端末3から、死活判定処理を行うサーバ2に送信される。対局画面の一例を図74に示す。

【0261】

対局画面が図74の場合、その画面のうち、「死活ナビ」など、死活判定処理を行うことを要求するボタンを押下することで、死活判定処理を行うことの要求が対局者端末3からサーバ2に送信される。

【0262】

サーバ2の入力受付部20でその要求を受け付けると、死活判定処理部21dは、対局システムの記憶装置11から当該盤面の情報（たとえば棋譜の情報）を取得する。そして盤面の情報が含まれた死活判定処理の画面を出力処理部22が対局者端末3に送信する。この際の画面の一例が図75である。なお実際に対局している盤面の情報のほか、すでに対局し、所定の記憶装置11に保存している棋譜の情報、あるいは観戦している対局の棋譜の情報を死活判定処理部21dが対局システムの記憶装置11から取得し、その棋譜の情報を図75の画面に含めて対局者端末3に送信しても良い。

【0263】

死活判定を行う対局者は、図75に示す死活判定処理を行う画面から盤面の編集を行う。例えば表示されている石を指定することで、石を取り除いたり、あるいは石を置く。またマウスなどの入力装置13をドラッグするなどにより、判定領域を指定する。このように死活判定処理を所望する対局者は、適宜、盤面の情報を編集する。この際の編集される石の情報、手番の情報は、対局者端末3からサーバ2に適宜、送られる。

【0264】

上述のように対局者が判定領域を指定することにより、その判定領域内にある結論を出すべき石が判定対象の石として設定される。また、対局者が判定対象の石のうち、さらに特定の石を結論を出すべき石（目標石）として設定する場合には、それを目標石として設定する。判定対象の石のうち少なくとも一つの石が活きる時には死活判定の結論は活きとなり、すべての石が取られる場合、結論は死となる。なお、目標石を指定した場合には、目標石のすべてが活きる時には死活判定の結論は活きとなり、目標石のうち少なくとも一つが取られる場合、死活判定の結論は死となる。なお目標石の指定は必須ではなく、また目標石を指定するかしないかではその処理には差がなく、活き死にの判断基準が変わるだけである。従って、目標石を指定した場合には、「判定対象の石」との記載を「目標石」と適宜読み替え、各グループ化された目標石についてそれぞれ処理を行うことで、その処理が可能となる。

【0265】

編集後の死活判定処理を行う画面の一例を図76に示す。なお図76では目標石が黒の2子（黒星）、手番は白であるとする。従って、この場合、次の順番は白石であり、さらに白石から見て黒の2子の死活判定を行うことを意味している。

10

20

30

40

50

【 0 2 6 6 】

例えば図 7 6 の画面における「判定処理」を押下することで、編集後の盤面の情報（棋譜の情報と判定対象の石の情報など）と手番の情報と、死活判定処理を開始することの要求とが対局者端末 3 からサーバ 2 に送信される。サーバ 2 の入力受付部 2 0 で編集後の盤面の情報と手番の情報と死活判定処理を開始することの要求を受け付けると、死活判定処理部 2 1 d は、入力を受け付けた編集後の盤面の情報に基づいて、死活判定処理を開始する。

【 0 2 6 7 】

まず死活判定処理部 2 1 d の E 空点カウント処理部 2 2 4 は、目標石である判定対象の石の E 空点の数をカウントする（S 1 4 0 0）。ここで E 空点の数が 1 点であれば（S 1 4 1 0）、簡易判定処理部 2 2 5 は攻撃側の勝ちであることを判定し（S 1 4 2 0）、そのまま処理を終了する。また E 空点の数が 4 以上であれば（S 1 4 1 0）、簡易判定処理部 2 2 5 は攻撃側の負けであることを判定し（S 1 4 3 0）、そのまま処理を終了する。

【 0 2 6 8 】

図 7 6 の場合、E 空点の数は図 7 7 に示すように 3 点ある。そのため、簡易判定処理部 2 2 5 は、各 E 空点について、上記の順序に従って、着手の優先順位を決定する（S 1 4 4 0）。ここでは、E 空点 1 が防御側が着手した時の E 空点が 5 カ所、E 空点 2 が防御側が着手した時の E 空点が 4 カ所、E 空点 3 が防御側が着手した時の E 空点が 3 カ所であるので、この優先順位に従って、着手の優先順位を決定する。

【 0 2 6 9 】

従って、簡易判定処理部 2 2 5 は、まず優先順位が最も高い E 空点 1 に着手する（S 1 4 5 0）。これを模式的に示すのが図 7 8 である。

【 0 2 7 0 】

次に、簡易判定処理部 2 2 5 は、この E 空点 1 に対して攻撃側が着手したことに対し、防御側のすべての着手で攻撃側が目標石を取れるかを判定する（S 1 4 6 0）。もし目標石が防御側のすべての着手で取れる場合には、その時点で簡易判定処理部 2 2 5 は処理を中止し、攻撃側の勝ちを判定する（S 1 4 9 0）。一方、目標石が防御側のいずれかの着手で取られない場合には（S 1 4 6 0）、S 1 4 5 0 で設定した優先順位の次の順位で着手を行う（S 1 4 8 0、S 1 4 5 0）。すなわち、図 7 7 の E 空点 2 に着手することとなる。これを模式的に示すのが、図 7 9 である。

【 0 2 7 1 】

そして、上記と同様に、簡易判定処理部 2 2 5 は、この E 空点 2 に対して攻撃側が着手したことに対し、防御側のすべての着手で攻撃側が目標石を取れるかを判定する（S 1 4 6 0）。もし目標石が防御側のすべての着手で取れる場合には、その時点で簡易判定処理部 2 2 5 は処理を中止し、攻撃側の勝ちを判定する（S 1 4 9 0）。一方、目標石が防御側のいずれかの着手で取られない場合には（S 1 4 6 0）、S 1 4 5 0 で設定した優先順位の次の順位で着手を行う（S 1 4 8 0、S 1 4 5 0）。すなわち、図 7 8 の E 空点 3 に着手することとなる。これを模式的に示すのが、図 8 0 である。

【 0 2 7 2 】

そして、上記と同様に、簡易判定処理部 2 2 5 は、この E 空点 3 に対して攻撃側が着手したことに対し、防御側のすべての着手で攻撃側が目標石を取れるかを判定する（S 1 4 6 0）。もし目標石が防御側のすべての着手で取れる場合には、その時点で簡易判定処理部 2 2 5 は処理を中止し、攻撃側の勝ちを判定する（S 1 4 9 0）。一方、目標石が防御側のいずれかの着手で取られない場合には（S 1 4 6 0）、S 1 4 5 0 で設定した優先順位の次の順位で着手を行うが（S 1 4 8 0）、すべての優先順位で着手を行ってしまったので、簡易判定処理部 2 2 5 は、攻撃側の負けを判定し、処理を終了する（S 1 4 9 0）。

【 0 2 7 3 】

以上のようにして死活判定処理部 2 1 d で死活判定の処理結果を判定すると、出力処理部 2 2 は、その処理結果を対局者端末 3 に送信する。

【 0 2 7 4 】

なお上述の S 1 4 6 0 で防御側のすべての着手で攻撃側が取れるか取れないかを判定するためには、防御側が E 空点カウント処理部 2 2 4 でカウントした E 空点の数に応じて着手した点に対し、さらに攻撃側が、E 空点カウント処理部 2 2 4 でカウントした E 空点の数に応じて着手する、といったように順に優先順位に従った処理を行う必要がある。そこで、以下に、図 7 8 の状態から、防御側の処理を説明する。

【 0 2 7 5 】

図 7 8 の状態に攻撃側が着手した場合、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は、2 カ所である。従って、第 1 に、E 点となる攻撃側の石の E 空点が 2 カ所の点、第 2 に、E 空点、第 3 に、O 空点、第 4 に、E 空点の E 空点、第 5 に、E 空点の O 空点、第 6 に O 空点の E 空点の優先順位を各点に判定することが出来る。これを模式的に示すのが、図 8 1 である。

10

【 0 2 7 6 】

この場合、優先順位が高い E 空点 1 が 2 点あるので、例えば上方にある E 空点 1 に着手する。これを模式的に示すのが図 8 2 である。

【 0 2 7 7 】

次に図 8 2 の状態に防御側が着手した場合、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は 2 点なので、上述の優先順位に従って着手する。すなわち、第 1 に、E 点となる攻撃側の石の E 空点が 1 カ所の点、第 2 に、E 空点、第 3 に、O 空点、第 4 に、E 空点の E 空点、第 5 に、E 空点の O 空点、第 6 に O 空点の E 空点の優先順位を、各点について判定することが出来る。これを模式的に示すのが、図 8 3 である。

20

【 0 2 7 8 】

そうすると優先順位が最も高いのが E 空点 1 なので、攻撃側はそこに着手することとなる。これを模式的に示すのが図 8 4 である。

【 0 2 7 9 】

次に図 8 4 の状態に攻撃側が着手した場合、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は 1 点なので、上述の防御側の優先順位に従って着手する。すなわち、E 空点が 1 点で、かつ E 点となる攻撃側の石に E 空点が 1 点となるものがないので、残りの E 空点に着手することとなる。これを模式的に示すのが図 8 5 である。

【 0 2 8 0 】

従って防御側はそこに着手することとなる。これを模式的に示すのが図 8 6 である。

30

【 0 2 8 1 】

次に図 8 6 の状態に防御側が着手した場合、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は 2 点なので、上述の優先順位に従って着手する。なお目標石と繋がっている同じ色についても目標石として処理を行うこととなる。すなわち、第 1 に、E 点となる防御側が着手したときの E 空点が 3 カ所の点、第 2 に、E 点となる防御側が着手したときの E 空点が 2 カ所の点、第 3 に、O 空点、第 4 に、E 空点の E 空点、第 5 に、E 空点の O 空点、第 6 に O 空点の E 空点の優先順位を、各点について判定することが出来る。これを模式的に示すのが、図 8 7 である。

【 0 2 8 2 】

そうすると優先順位が最も高いのが E 空点 1 なので、攻撃側はそこに着手することとなる。攻撃側が着手した状態を模式的に示すのが図 8 8 である。

40

【 0 2 8 3 】

次に図 8 8 の状態に攻撃側が着手した場合、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は 1 点なので、上述の優先順位に従って着手する。すなわち、E 空点が 1 点で、かつ E 点となる攻撃側の石に E 空点が 1 点となるものがないので、残りの E 空点に着手することとなる。これを模式的に示すのが図 8 9 である。

【 0 2 8 4 】

従って防御側はそこに着手することとなる。これを模式的に示すのが図 9 0 である。

【 0 2 8 5 】

50

次に図 9 0 の状態に防御側が着手した場合、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は 1 点なので、上述の優先順位に従って着手する。これを模式的に示すのが図 9 1 である。すなわち、E 空点 1 に着手するとともに、攻撃側の勝ちを判定する。これを模式的に示すのが図 9 2 である。

【 0 2 8 6 】

以上のように行うことで攻撃側の勝ちを判定できるが、これは図 8 1 の状態において、優先順位が「 1 」である 2 点のうち、上方にある E 空点 1 に着手した場合である（これが図 8 2 である）。従ってもう一つの優先順位が「 1 」である点については、攻撃側の勝ちを簡易判定処理部 2 2 5 は判定できていないので、もう一つの点について着手した場合を、簡易判定処理部 2 2 5 は判定する。すなわち、一度、盤面の状態を図 8 1 の状態に戻した上で、もう一つの優先順位が「 1 」である点に着手する。これを模式的に示すのが図 9 3 である。

【 0 2 8 7 】

次に図 9 3 の状態に防御側が着手した場合、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は 2 点なので、上述の優先順位に従って着手する。すなわち、第 1 に、E 点となる攻撃側の石の E 空点が 1 カ所の点、第 2 に、E 点となる防御側が着手したときの E 空点が 2 カ所の点、第 2 に、E 点となる防御側が着手したときの E 空点が 1 カ所の点、第 3 に、O 空点、第 4 に、E 空点の E 空点、第 5 に、E 空点の O 空点、第 6 に O 空点の E 空点の優先順位を、各点について判定することが出来る。これを模式的に示すのが、図 9 4 である。

【 0 2 8 8 】

そうすると優先順位が最も高いのが E 空点 1 なので、攻撃側はそこに着手することとなる。攻撃側が着手した状態を模式的に示すのが図 9 5 である。

【 0 2 8 9 】

次に図 9 5 の状態に攻撃側が着手した場合、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は 2 点なので、上述の優先順位に従って着手する。すなわち、第 1 に、E 点のうち防御側が着手したときの E 空点が 2 カ所の点、第 2 に、O 空点、第 3 に、E 空点の E 空点、第 4 に、E 空点の O 空点、第 5 に、O 空点の E 空点の優先順位を、各点について判定することが出来る。これを模式的に示すのが図 9 6 である。

【 0 2 9 0 】

図 9 6 の状態では優先順位が「 1 」である点（E 空点 1）が 2 点ある。そこでまず E 空点 1 のうち、上方の E 空点 1 に防御側は着手する。これを模式的に示すのが図 9 7 である。なお下方の E 空点 1 についても、上方の E 空点 1 に着手した場合の処理が終了したあとで着手し、同様に判定する。

【 0 2 9 1 】

次に図 9 7 の状態に防御側が着手した場合、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は 2 点なので、上述の優先順位に従って着手する。すなわち、第 1 に、E 点となる防御側が着手したときの E 空点が 3 カ所の点、第 2 に、E 点となる防御側が着手したときの E 空点が 2 カ所の点、第 3 に、O 空点、第 4 に、E 空点の E 空点、第 5 に、E 空点の O 空点、第 6 に O 空点の E 空点の優先順位を、各点について判定することが出来る。これを模式的に示すのが、図 9 8 である。

【 0 2 9 2 】

そうすると優先順位が最も高いのが E 空点 1 なので、攻撃側はそこに着手することとなる。攻撃側が着手した状態を模式的に示すのが図 9 9 である。

【 0 2 9 3 】

次に図 9 9 の状態に攻撃側が着手した場合、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は 1 点なので、上述の優先順位に従って着手する。すなわち、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は 1 点なので、上述の優先順位に従って着手する。すなわち、E 空点が 1 点で、かつ E 点となる攻撃側の石に E 空点が 1 点となるものがないので、残りの E 空点に着手することとなる。これを模式的に示すのが図 1 0 0 である

10

20

30

40

50

。

【 0 2 9 4 】

従って防御側はそこに着手することとなる。これを模式的に示すのが図 1 0 1 である。

【 0 2 9 5 】

次に図 1 0 1 の状態に防御側が着手した場合、E 空点カウント処理部 2 2 4 がカウントする E 空点の数は 1 点なので、上述の優先順位に従って着手する。これを模式的に示すのが図 1 0 2 である。すなわち、E 空点 1 に着手するとともに、攻撃側の勝ちを判定する。これを模式的に示すのが図 1 0 3 である。

【 0 2 9 6 】

以上のように行うことで攻撃側の勝ちを判定できるが、これは図 9 6 の状態において、優先順位が「1」である 2 点のうち、上方にある E 空点 1 に着手した場合である（これが図 9 7 である）。従ってもう一つの優先順位が「1」である点については、攻撃側の勝ちを簡易判定処理部 2 2 5 は判定できていないので、もう一つの点について着手した場合を、簡易判定処理部 2 2 5 は判定する。すなわち、一度、盤面の状態を図 9 6 の状態に戻した上で、もう一つの優先順位が「1」である点に着手する。これを模式的に示すのが図 1 0 4 である。

【 0 2 9 7 】

上述の処理を簡易判定処理部 2 2 5 は、攻撃側、防御側についてそれぞれ E 空点の数に応じた着手を繰り返す。そして、簡易判定処理部 2 2 5 は、攻撃側、防御側のいずれについても優先順位が高い順にそれぞれ着手を行う。その結果、攻撃側の特定の着手について攻撃側で取れる場合は攻撃側の勝ち、攻撃側のすべての着手で取れないような防御側の着手がある場合には攻撃側の負けと判定する。

【 0 2 9 8 】

このような処理を用いることで、特に終局状態の盤面で特定の石、すなわち目標石が取られるかを高速に判定することが出来る。

【 0 2 9 9 】

次に上述の各処理のより詳細な説明を図 1 0 5 乃至図 1 1 6 を用いて説明する。

【 0 3 0 0 】

まず死活判定処理部 2 1 d は、盤面の情報を読み込み、Board[x][y] に空点であるか、黒石が置かれているか、白石が置かれているかを記憶していく。また判定領域 Area[x][y] にそれぞれ判定領域であるか否かを記憶していく。また上記のほか、判定対象の石 H[x][y]、判定領域 Area[x][y]、手番 t n など適宜、設定される。そして、死活判定処理部 2 1 d における E 空点カウント処理部 2 2 4 が、Board[x][y]、H[x][y] から、H[x][y] の E 空点を取得し、配列 E 1[x][y] に代入する (S 1 5 0 0)。

【 0 3 0 1 】

そして簡易判定処理部 2 2 5 は、配列 E 1[x][y] = 1 となる点の数に応じて、S 1 5 2 0 乃至 S 1 5 5 0 に応じた処理を行う (S 1 5 1 0)。

【 0 3 0 2 】

まず E 1[x][y] = 1 となる点の数が 4 以上の場合、簡易判定処理部 2 2 5 は、攻撃側の負けであることを判定し (S 1 5 2 0)、そのまま処理を終了する。また、E 1[x][y] = 1 となる点の数が 1 である場合、簡易判定処理部 2 2 5 は、攻撃側の勝ちであることを判定し (S 1 5 3 0)、そのまま処理を終了する。

【 0 3 0 3 】

また E 1[x][y] = 1 となる点の数が 3 の場合、簡易判定処理部 2 2 5 は、処理 B を実行し (S 1 5 4 0)、処理 B が終了後、さらに処理 D を実行する (S 1 5 6 0)。

【 0 3 0 4 】

一方、E 1[x][y] = 1 となる点の数が 2 の場合、簡易判定処理部 2 2 5 は、処理 C を実行し (S 1 5 5 0)、処理 C が終了後、さらに処理 D を実行する (S 1 5 6 0)。

【 0 3 0 5 】

10

20

30

40

50

以上のような処理を簡易判定処理部 225 は実行する。

【0306】

次に、まず処理 B を説明する。簡易判定処理部 225 は、 $E1[x][y] = 1$ となる点 (x, y) について S1600 乃至 S1635 を反復処理する。

【0307】

すなわち、まず $E1[x][y] = 1$ となる点 (x, y) について $Board[x][y] = tn^3$ を代入する (S1600)。そして配列 $Board[x][y]$ についてグループ化し、配列 $H2[x][y]$ に代入する (S1610)。次に、配列 $H2[x][y]$ の E 空点を取得し、配列 $E2[x][y]$ に代入する (S1620)。そして、その点 (x, y) を配列 $Order[i]$ に追加し (配列 $Order[i].x$ に x を、配列 $Order[i].y$ に y を代入)、 $Order[i].v$ に $E2[x][y] = 1$ となる点の数を代入する (S1630)。また配列 $Board[x][y] = 0$ を代入する (S1635)。

【0308】

以上の処理を $E1[x][y] = 1$ となるすべての点 (x, y) について処理を行い、処理終了後、配列 $Order$ を、その要素 v が大きい順に並べ替える (S1640)。

【0309】

以上のようにして簡易判定処理部 225 は処理 B を実行し、その後、処理 D を実行する。

【0310】

まず $i = 0$ から $i < 361$ になるまで、S1650 乃至 S1660 を反復処理する。

【0311】

すなわち、まず手番 tn で配列 $Order[i]$ の点 (x, y) に着手可能かを判定し (S1650)、着手不可能ならば次の i の処理に移る。一方、着手可能ならば、着手を行い、配列 $Board[x][y]$ 、配列 $H[x][y]$ を更新する (S1655)。すなわち $Board[x][y] = tn$ とし、さらに着手したことによって相手の石が取れる場合には、取れる石の位置における配列 $Board$ の値についても適宜更新する。また判定対象の石と隣接している石についても配列 $H[x][y] = 1$ として適宜更新する。さらにグループ化できる場合には、配列 $H[x][y]$ を再グループ化する。その後、後述する処理 E を実行し、処理 E の判定結果が勝ちならば (S1660)、簡易判定処理部 225 は攻撃側の勝ちと判定し (S1670)、処理を終了する。一方、処理 E の判定結果が負けならば次の i の処理に移る。

【0312】

以上の処理を $i < 361$ となるまで反復し、処理を終了した場合、簡易判定処理部 225 は攻撃側の負けと判定し (S1680)、処理を終了する。

【0313】

次に、図 105 の S1550 における処理 C を説明する。

【0314】

簡易判定処理部 225 は、 $H[x][y]$ の空点ではない E 点について、配列 $E2[x][y]$ に代入する (S1700)。そして、 $E2[x][y] = 1$ となるすべての点 (x, y) について、S1710 乃至 S1740 を反復処理する。

【0315】

すなわち、まず $Board[x][y]$ をグループ化し、配列 $G[x][y]$ に代入する (S1710)。そして配列 $G[x][y]$ の E 空点を配列 $G2[x][y]$ に代入する (S1720)。そして配列 $G2[a][b] = 1$ ($1 \leq a, b \leq 19$) となる点 (a, b) の数を判定し (S1730)、それが 1 点以外の場合には、次の点 (x, y) に移る。一方、配列 $G2[a][b] = 1$ となる点の数が 1 点の場合には、点 (a, b) を配列 $Order[i]$ に追加し (配列 $Order[i].x$ に a を、配列 $Order[i].y$ に b を代入)、 $Order[i].v = 5000$ とする (S1740)。

【0316】

以上の処理を $E2[x][y] = 1$ となるすべての点 (x, y) について行い、その処理終了後、処理 C2 を実行する。

【0317】

まず $E1[x][y] = 1$ となるすべての点 (x, y) について、S1750乃至S1790を反復処理する。

【0318】

すなわち、 $E1[x][y] = 1$ となる点 (x, y) について $Board[x][y] = tn^3$ を代入する (S1750)。そして $Board[x][y]$ をグループ化して、配列 $H2[x][y]$ に代入する (S1760)。次に配列 $H2$ の E 空点を配列 $E2[x][y]$ に代入する (S1770)。そして点 (x, y) を配列 $Order[i]$ に追加し (配列 $Order[i].x$ に x を、配列 $Order[i].y$ に y を代入)、 $Order[i].v$ に、4000と $E2[x][y] = 1$ となる点の数を加算した値を代入する (S1780)。そして $Board[x][y] = 0$ を代入する (S1790)。

【0319】

以上の処理を $E1[x][y] = 1$ となるすべての点 (x, y) について行い、その処理終了後、処理 C3 を実行する。

【0320】

まず、配列 $H[x][y]$ の O 空点を配列 $O[x][y]$ に代入する (S1800)。そして配列 $H[x][y]$ の E 空点を配列 $EE[x][y]$ に代入する (S1810)。次に配列 $H[x][y]$ の E 空点の O 空点を配列 $EO[x][y]$ に代入し (S1820)、配列 $H[x][y]$ の O 空点の E 空点を配列 $OE[x][y]$ に代入する (S1830)。

【0321】

そして点 (x, y) について、それぞれ 1 から 19 になるまで、S1840乃至S1910を反復処理する。

【0322】

まず配列 $O[x][y] = 1$ であるかを判定し (S1840)、そうであるならば点 (x, y) を $Order[i]$ に追加して (配列 $Order[i].x$ に x を、配列 $Order[i].y$ に y を代入)、 $Order[i].v = 3000$ を代入する (S1850)。そして次の点 (x, y) の処理に移る。また、配列 $O[x][y] = 1$ ではない場合には、配列 $EE[x][y] = 1$ であるかを判定する (S1860)。

【0323】

配列 $EE[x][y] = 1$ であるならば点 (x, y) を配列 $Order[i]$ に追加して (配列 $Order[i].x$ に x を、配列 $Order[i].y$ に y を代入)、 $Order[i].v = 2000$ を代入する (S1870)。そして次の点 (x, y) の処理に移る。そうでないならば、配列 $EO[x][y] = 1$ であるかを判定する (S1880)。

【0324】

配列 $EO[x][y] = 1$ であるならば点 (x, y) を配列 $Order[i]$ に追加して (配列 $Order[i].x$ に x を、配列 $Order[i].y$ に y を代入)、 $Order[i].v = 1000$ を代入する (S1890)。そして次の点 (x, y) の処理に移る。そうでないならば、配列 $OE[x][y] = 1$ であるかを判定する (S1900)。

【0325】

配列 $OE[x][y] = 1$ であるならば点 (x, y) を配列 $Order[i]$ に追加して (配列 $Order[i].x$ に x を、配列 $Order[i].y$ に y を代入)、 $Order[i].v = 0$ を代入する (S1910)。そして次の点 (x, y) の処理に移る。またそうでない場合にも次の点 (x, y) の処理に移る。

【0326】

以上の処理を点 (x, y) がそれぞれ 19 になるまで処理を実行し、処理終了後、配列

Order [i] について、その要素 v が大きい順に並べ替える (S 1 9 2 0)。そして、上述の処理 D を簡易判定処理部 2 2 5 が実行する。

【 0 3 2 7 】

次に処理 E について説明する。

【 0 3 2 8 】

まず処理 E においては、配列 H [x] [y] の E 空点を配列 E 1 [x] [y] に代入する (S 1 9 5 0)。そして $E 1 [x] [y] = 1$ となる点の数を判定し (S 1 9 6 0)、それが 2 であるならば、後述する処理 F を簡易判定処理部 2 2 5 は実行する (S 1 9 7 0)。また 1 であるならば、後述する処理 G を簡易判定処理部 2 2 5 は実行する (S 1 9 8 0)。そして処理 F、処理 G が終了後、簡易判定処理部 2 2 5 は後述する処理 H を実行する (S 1 9 9 0)。

10

【 0 3 2 9 】

まず $E 1 [x] [y] = 1$ となる点の数が 2 であった場合に実行する処理 F について説明する。

【 0 3 3 0 】

処理 F では、まず配列 H [x] [y] の空点ではない E 点を配列 E 2 [x] [y] に代入する (S 2 0 0 0)。そして $E 2 [x] [y] = 1$ となるすべての点 (x , y) について、S 2 0 1 0 乃至 S 2 0 5 0 を反復処理する。

【 0 3 3 1 】

まず配列 Board [x] [y] をグループ化し、配列 G [x] [y] に代入する (S 2 0 1 0)。そして配列 G [x] [y] の E 空点を配列 G 2 [x] [y] に代入する (S 2 0 2 0)。

20

【 0 3 3 2 】

そして配列 $G 2 [a] [b] = 1$ (1 a , b 1 9) となる点 (a , b) の数を判定し (S 2 0 3 0)、それが 1 であるならば、点 (a , b) を配列 Order [i] に追加し (配列 Order [i] . x に a を、配列 Order [i] . y に b を代入)、Order [i] . v = 6 0 0 0 を代入する (S 2 0 4 0)。一方、配列 $G 2 [a] [b] = 1$ となる点の数が 2 の場合、点 (a , b) を配列 Order [i] に追加し (配列 Order [i] . x に a を、配列 Order [i] . y に b を代入)、Order [i] . v = 5 0 0 0 を代入する (S 2 0 5 0)。また配列 $G 2 [a] [b]$ が 1 でも 2 でもない場合には、あるいは S 2 0 4 0、S 2 0 5 0 において点 (a , b) を配列 Order [i] に追加した場合には、次の点 (x , y) の処理に移る。

30

【 0 3 3 3 】

以上の処理を $E 2 [x] [y] = 1$ となるすべての点 (x , y) について実行し、処理終了後、処理 F 2 を実行する。

【 0 3 3 4 】

まず、配列 $E 1 [x] [y] = 1$ となるすべての点 (x , y) について、S 2 0 6 0 乃至 S 2 1 0 0 の反復処理を実行する。

【 0 3 3 5 】

まず、 $Board [x] [y] = t n ^ 3$ を代入し (S 2 0 6 0)、次に Board [x] [y] をグループ化して、配列 H 2 [x] [y] に代入する (S 2 0 7 0)。そして配列 H 2 [x] [y] の E 空点を配列 E 2 [x] [y] に代入し (S 2 0 8 0)、点 (x , y) を配列 Order [i] に追加する (配列 Order [i] . x に x を、配列 Order [i] . y に y を代入)。そして Order [i] . v について、4 0 0 0 と、配列 $E 2 [x] [y] = 1$ となる点の数とを加算した値を代入する (S 2 0 9 0)。そして、 $Board [x] [y] = 0$ を代入する (S 2 1 0 0)。

40

【 0 3 3 6 】

以上の処理を $E 1 [x] [y] = 1$ となるすべての点 (x , y) について実行し、処理終了後、処理 F 3 を実行する。

【 0 3 3 7 】

50

まず、配列 $H[x][y]$ の O 空点を配列 $O[x][y]$ に代入する (S 2 1 1 0)。そして配列 $H[x][y]$ の E 空点を配列 $EE[x][y]$ に代入する (S 2 1 2 0)。次に配列 $H[x][y]$ の E 空点の O 空点を配列 $EO[x][y]$ に代入し (S 2 1 3 0)、配列 $H[x][y]$ の O 空点の E 空点を配列 $OE[x][y]$ に代入する (S 2 1 4 0)。

【0338】

そして点 (x, y) について、それぞれ 1 から 19 になるまで、S 2 1 5 0 乃至 S 2 2 2 0 を反復処理する。

【0339】

まず配列 $O[x][y] = 1$ であるかを判定し (S 2 1 5 0)、そうであるならば点 (x, y) を $Order[i]$ に追加して (配列 $Order[i].x$ に x を、配列 $Order[i].y$ に y を代入)、 $Order[i].v = 3000$ を代入する (S 2 1 6 0)。そして次の点 (x, y) の処理に移る。また、配列 $O[x][y] = 1$ ではない場合には、配列 $EE[x][y] = 1$ であるかを判定する (S 2 1 7 0)。

【0340】

配列 $EE[x][y] = 1$ であるならば点 (x, y) を配列 $Order[i]$ に追加して (配列 $Order[i].x$ に x を、配列 $Order[i].y$ に y を代入)、 $Order[i].v = 2000$ を代入する (S 2 1 8 0)。そして次の点 (x, y) の処理に移る。そうでないならば、配列 $EO[x][y] = 1$ であるかを判定する (S 2 1 9 0)。

【0341】

配列 $EO[x][y] = 1$ であるならば点 (x, y) を配列 $Order[i]$ に追加して (配列 $Order[i].x$ に x を、配列 $Order[i].y$ に y を代入)、 $Order[i].v = 1000$ を代入する (S 2 2 0 0)。そして次の点 (x, y) の処理に移る。そうでないならば、配列 $OE[x][y] = 1$ であるかを判定する (S 2 2 1 0)。

【0342】

配列 $OE[x][y] = 1$ であるならば点 (x, y) を配列 $Order[i]$ に追加して (配列 $Order[i].x$ に x を、配列 $Order[i].y$ に y を代入)、 $Order[i].v = 0$ を代入する (S 2 2 2 0)。そして次の点 (x, y) の処理に移る。またそうでない場合にも次の点 (x, y) の処理に移る。

【0343】

以上の処理を点 (x, y) がそれぞれ 19 になるまで処理を実行し、処理終了後、配列 $Order[i]$ について、その要素 v が大きい順に並べ替える (S 2 2 3 0)。そして、後述する処理 H を簡易判定処理部 225 が実行する。

【0344】

また、図 111 の S 1960 において配列 $E1[x][y] = 1$ となる点の数が 1 である場合には、処理 G が実行されるので、処理 G を説明する。

【0345】

簡易判定処理部 225 は、配列 $H[x][y]$ の空点ではない E 点について、配列 $E2[x][y]$ に代入する (S 2 3 0 0)。そして、 $E2[x][y] = 1$ となるすべての点 (x, y) について、S 2 3 1 0 乃至 S 2 3 4 0 を反復処理する。

【0346】

すなわち、まず点 (x, y) をグループ化し (配列 $Board[x][y]$ をグループ化し)、配列 $G[x][y]$ に代入する (S 2 3 1 0)。そして配列 $G[x][y]$ の E 空点を配列 $G2[x][y]$ に代入する (S 2 3 2 0)。そして配列 $G2[a][b] = 1$ ($1 \leq a, b \leq 19$) となる点 (a, b) の数を判定し (S 2 3 3 0)、それが 1 点以外の場合には、次の点 (x, y) に移る。一方、配列 $G2[x][y] = 1$ となる点の数が 1 点の場合には点 (a, b) を配列 $Order[i]$ に追加し (配列 $Order[i].x$ に a を、配列 $Order[i].y$ に b を代入)、 $Order[i].v = 5000$

10

20

30

40

50

とする (S 2 3 4 0)。

【 0 3 4 7 】

以上の処理を $E 2 [x] [y] = 1$ となるすべての点 (x, y) について行い、その処理終了後、処理 H を実行する。

【 0 3 4 8 】

まず簡易判定処理部 2 2 5 は、 $i = 0$ から $i < 3 6 1$ になるまで、S 2 4 0 0 乃至 S 2 4 3 0 を反復処理する。

【 0 3 4 9 】

すなわち、まず手番 $t n ^ 3$ で配列 $O r d e r [i]$ の点 (x, y) に着手可能かを判定し (S 2 4 0 0)、着手不可能ならば次の i の処理に移る。一方、着手可能ならば、着手を行い、配列 $B o a r d [x] [y]$ 、配列 $H [x] [y]$ を更新する (S 2 4 0 5)。すなわち $B o a r d [x] [y] = t n ^ 3$ とし、さらに着手したことによって相手の石が取れる場合には、取れる石の位置における配列 $B o a r d$ の値についても適宜更新する。また判定対象の石と隣接している石についても配列 $H [x] [y] = 1$ として適宜更新する。さらにグループ化できる場合には、配列 $H [x] [y]$ を再グループ化する。その後、後述する処理 A (図 1 0 5 の処理) を実行し、処理 A の判定結果が負けならば (S 2 4 1 0)、簡易判定処理部 2 2 5 は攻撃側の負けと判定し (S 2 4 2 0)、処理を終了する。一方、処理 A の判定結果が負けでないならば次の i の処理に移る。

【 0 3 5 0 】

以上の処理を $i < 3 6 1$ となるまで反復し、処理を終了した場合、簡易判定処理部 2 2 5 は攻撃側の勝ちと判定し (S 2 4 3 0)、処理を終了する。

【 0 3 5 1 】

以上のように、簡易判定処理部 2 2 5 が処理 A 乃至処理 H を実行することで、本実施例における、緩みシチョウを用いた死活判定処理を実行することが出来る。

【実施例 5】

【 0 3 5 2 】

なお上述の実施例 1 乃至実施例 3 では判定対象の石を用いた場合を説明したが、目標石を指定しても良い。この場合、死活判定処理部 2 1 d は、総あたりによる死活判定処理を行う場合に、目標石のすべてが活きる場合、死活判定の結果は生きとなり、目標石の少なくとも一つの石が取られる場合、死活判定の結果は死にと判定することで、総あたりによる死活判定処理を実行することが出来る。

【 0 3 5 3 】

また、上述の各実施例では配列、変数などを用いた処理を説明したが、それらに限定されず、適宜、必要な方法で処理を行うことが出来る。

【 0 3 5 4 】

本発明に於ける各手段、データベースは、その機能が論理的に区別されているのみであって、物理上あるいは事実上は同一の領域を為していても良い。又データベースの代わりにデータファイルであっても良いことは言うまでもなく、データベースとの記載にはデータファイルをも含んでいる。

【産業上の利用可能性】

【 0 3 5 5 】

上述の囲碁の死活判定システム 1 を用いることによって、対局中、終局時に限られず、また従来の死活判定システム 1 よりも高速に死活判定の結果を出力することが出来る。

【符号の説明】

【 0 3 5 6 】

1 : 囲碁の死活判定システム

2 : サーバ

3 : 対局者端末

1 0 : 演算装置

1 1 : 記憶装置

10

20

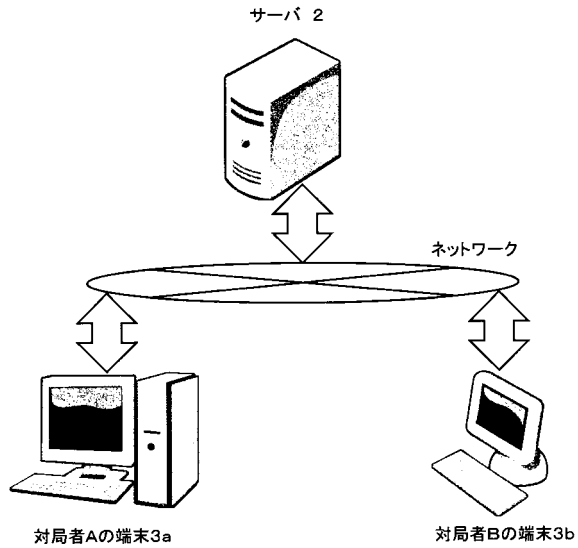
30

40

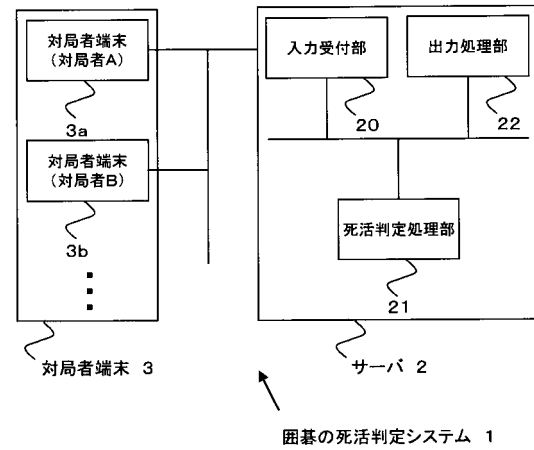
50

1 2 : 表示装置	
1 3 : 入力装置	
1 4 : 通信装置	
2 0 : 入力受付部	
2 1 : 死活判定処理部	
2 1 a、2 1 b、2 1 c、2 1 d : 死活判定処理部	
2 2 : 出力処理部	
2 1 0 : 初期判定処理部	
2 1 1 : 第 1 判定処理部	
2 1 2 : 第 2 判定処理部	10
2 1 3 : 優先度変更処理部	
2 1 4 : 死活判定実行部	
2 1 5 : 優先度記憶部	
2 1 6 : 第 1 外部処理部	
2 1 7 : 第 2 外部処理部	
2 1 8 : 死活判定実行部	
2 1 9 : 領域分割処理部	
2 2 0 : 0 眼変形処理部	
2 2 1 : 後手 1 眼変形処理部	
2 2 2 : 1 眼変形処理部	20
2 2 3 : 死活判定実行部	
2 2 4 : E 空点カウント処理部	
2 2 5 : 簡易判定処理部	
【要約】	
【課題】	
コンピュータ上で囲碁の対局ゲームを行う場合の、囲碁の死活判定システムを提供することを目的とする。	
【解決手段】	
コンピュータで囲碁の死活判定を行う、囲碁の死活判定システムであって、死活判定システムは、対局者による入力を受け付ける入力受付部と、判定対象の石または目標石の死活判定を行う死活判定処理部と、死活判定処理部における死活判定の結果を出力する出力処理部と、を備えており、死活判定処理部は、予め定められた判定領域内に、判定対象の石と異なる色の石を所定条件に従って配置し、配置した後の盤面の状態において死活判定処理を行う、囲碁の死活判定システムである。	30
【選択図】 図 1	

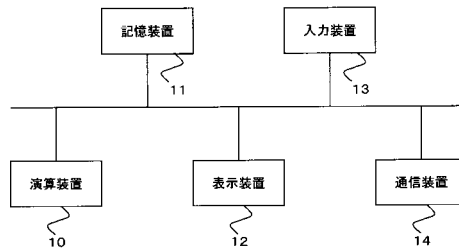
【図 1】



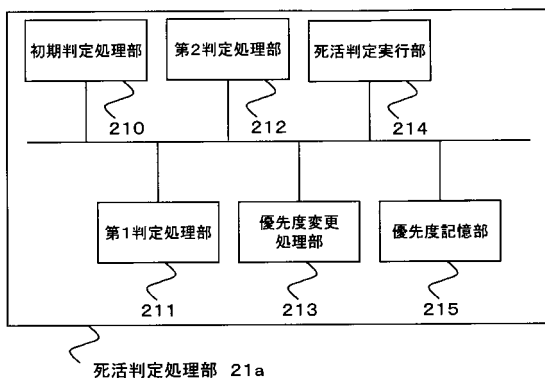
【図 2】



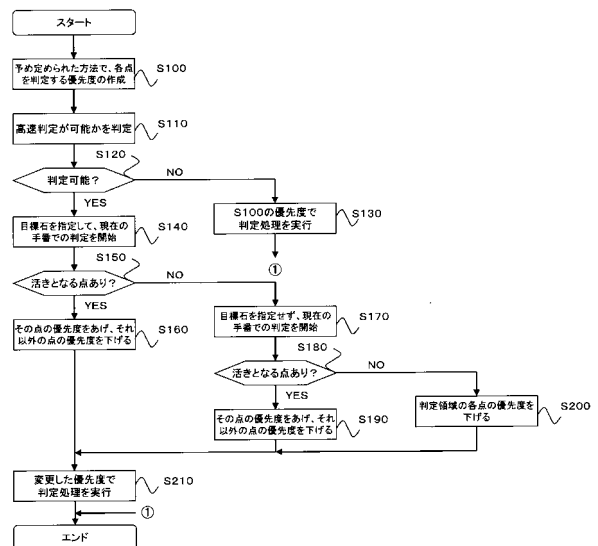
【図 3】



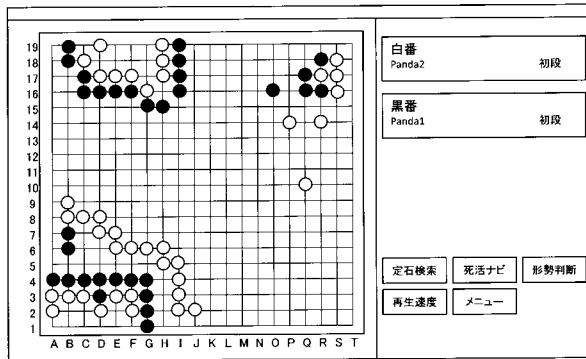
【図 4】



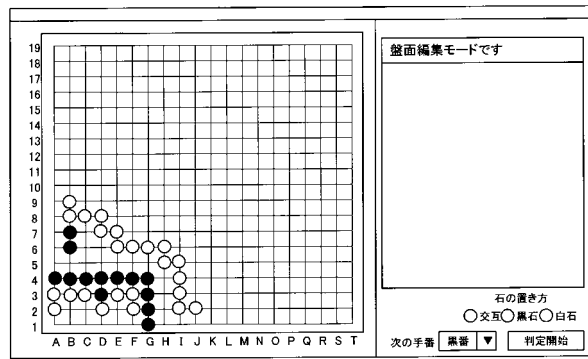
【図 5】



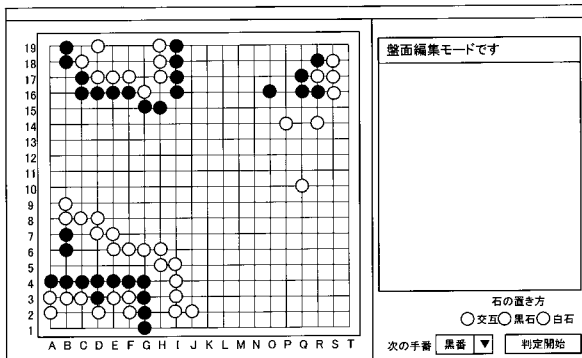
【図 6】



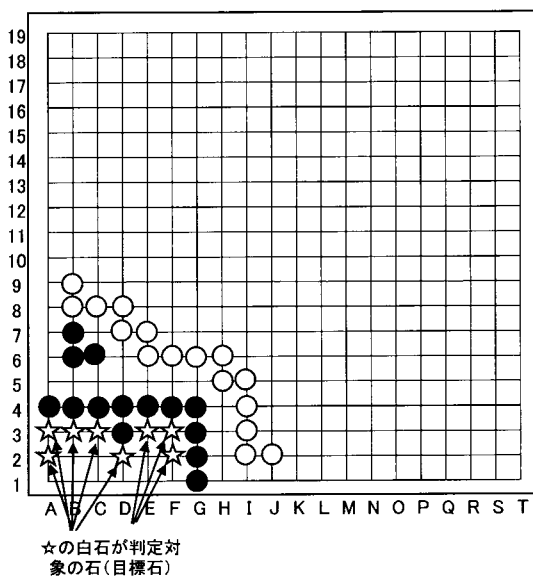
【図 8】



【図 7】

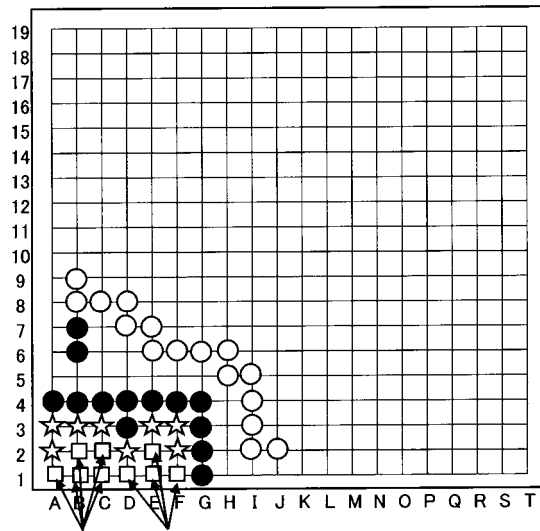


【図 9】



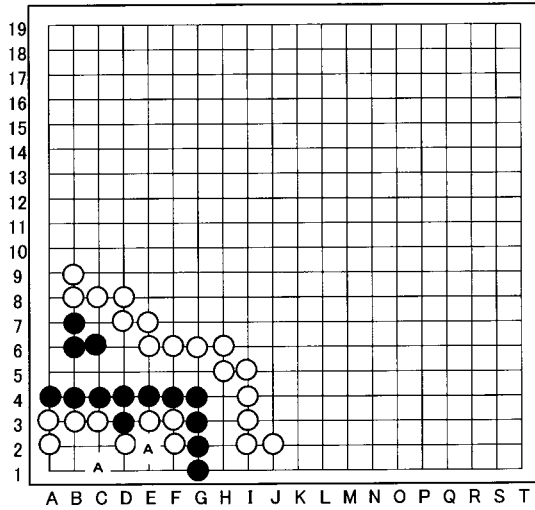
☆の白石が判定対象の石(目標石)

【図 10】

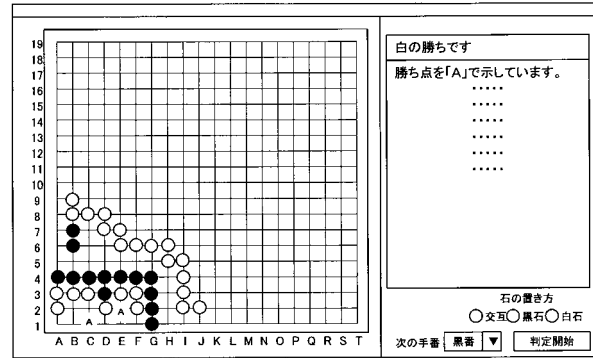


総あたりの対象となる判定領域内の空点

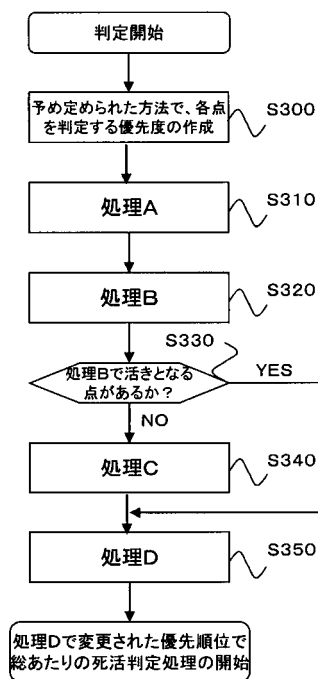
【 図 1 1 】



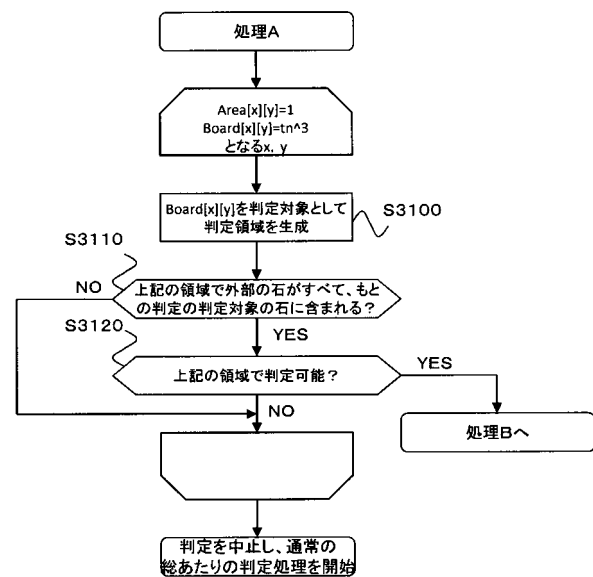
【 図 1 2 】



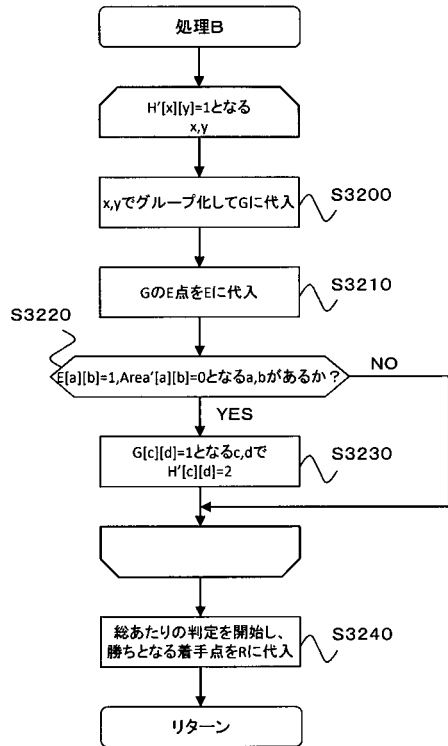
【 図 1 3 】



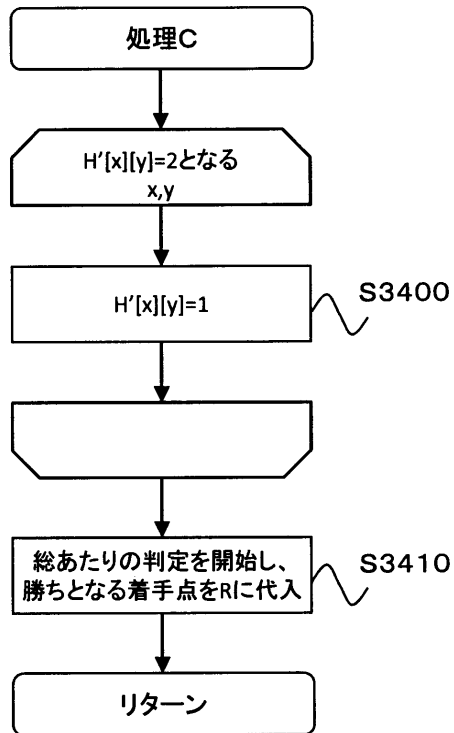
【 図 1 4 】



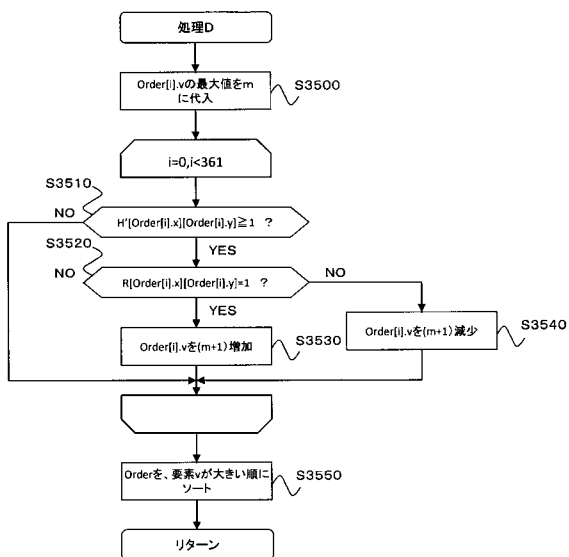
【図 15】



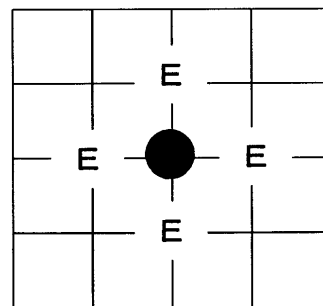
【図 16】



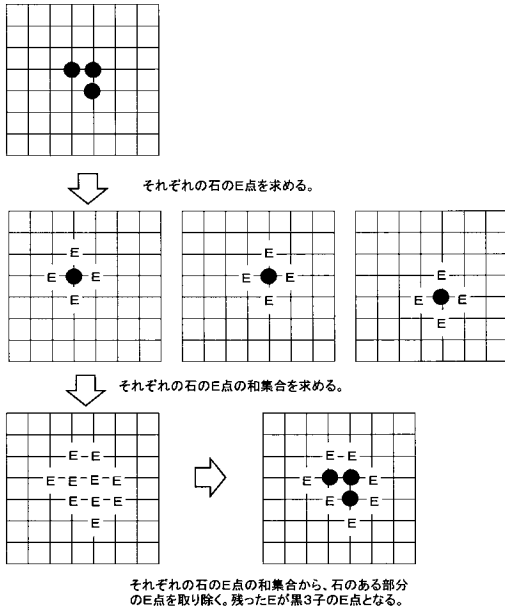
【図 17】



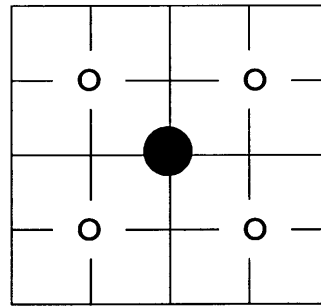
【図 18】



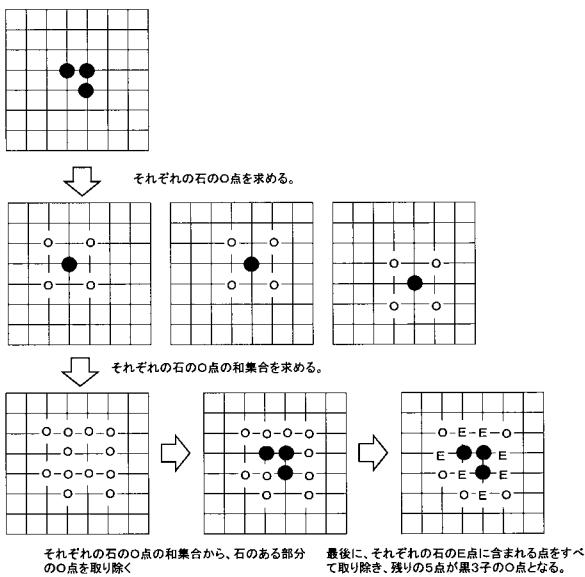
【図 19】



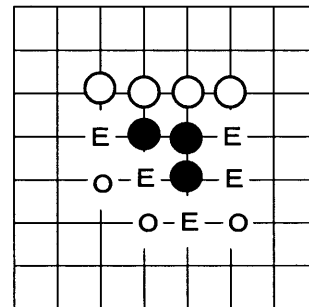
【図 20】



【図 21】

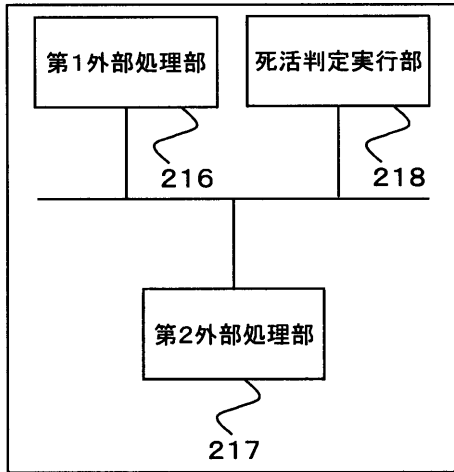


【図 22】



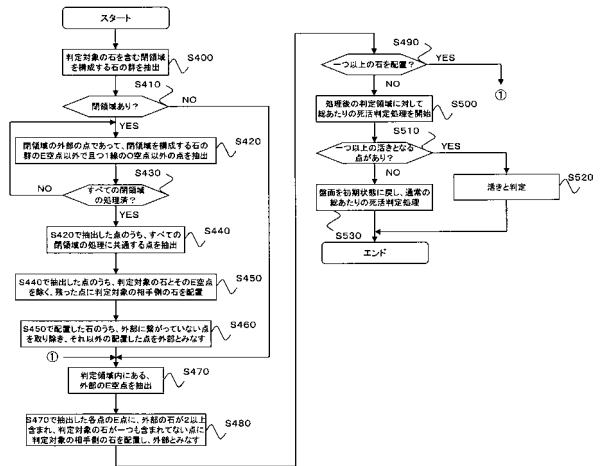
黒3子のE空点、O空点は、それぞれE、Oとなる。

【図 23】

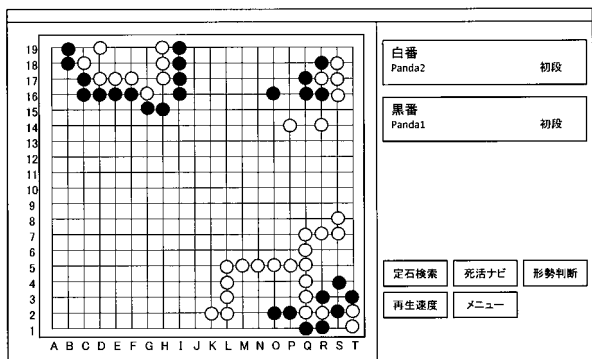


死活判定処理部 21b

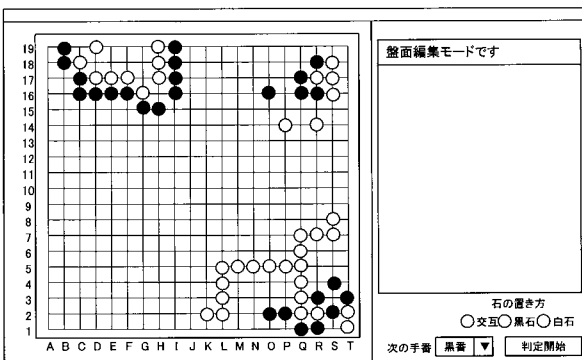
【図 24】



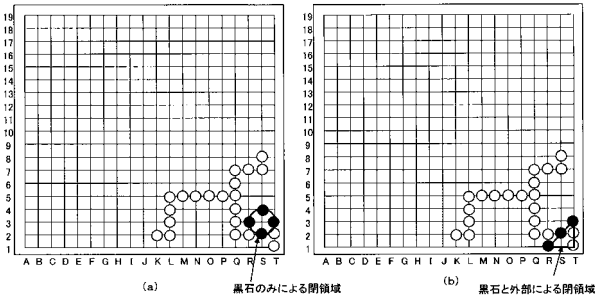
【図 25】



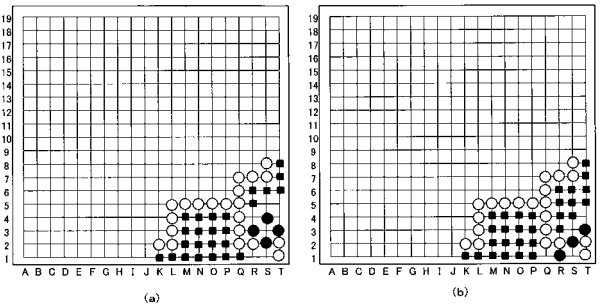
【図 26】



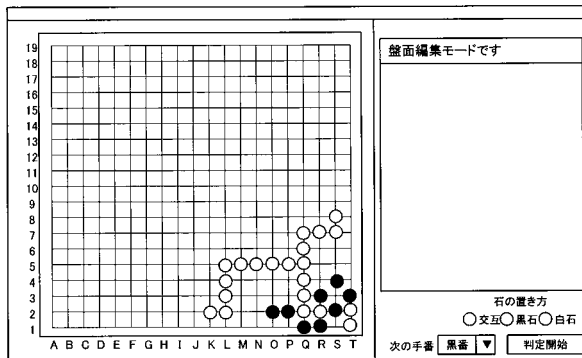
【図 28】



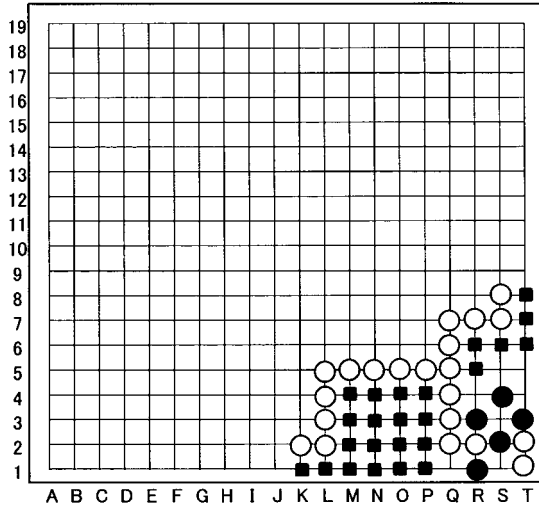
【図 29】



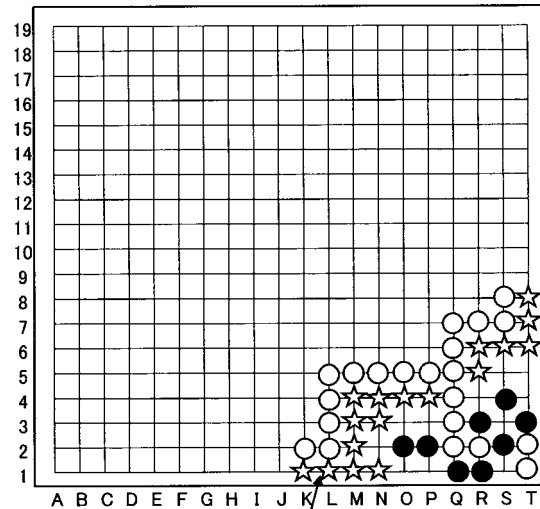
【図 27】



【図 30】

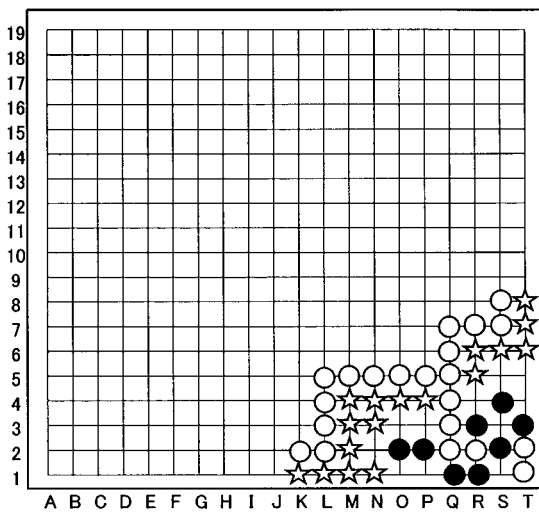


【図 31】



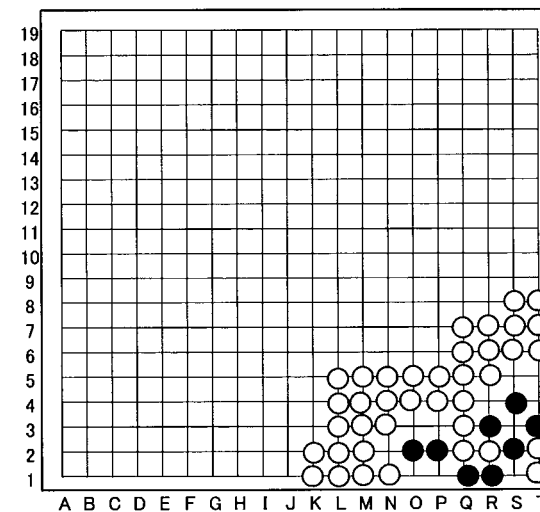
新たに配置される相手側の石(白石)

【図 32】

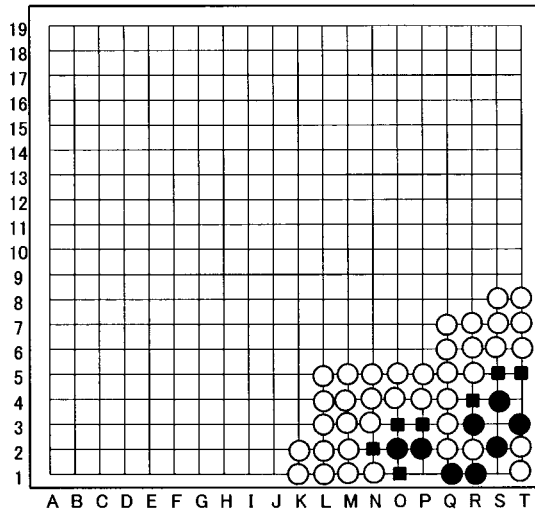


新たに配置される相手側の石(白石)から、外部に繋がっていない石を取り除く。
※ただし図31の場合、取り除かれる点はない。

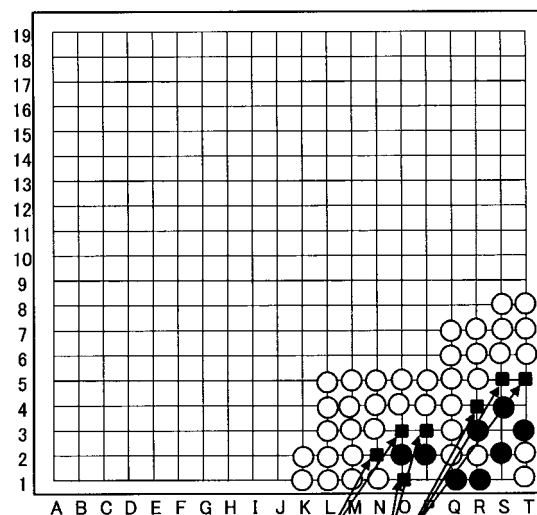
【図 33】



【図 3 4】

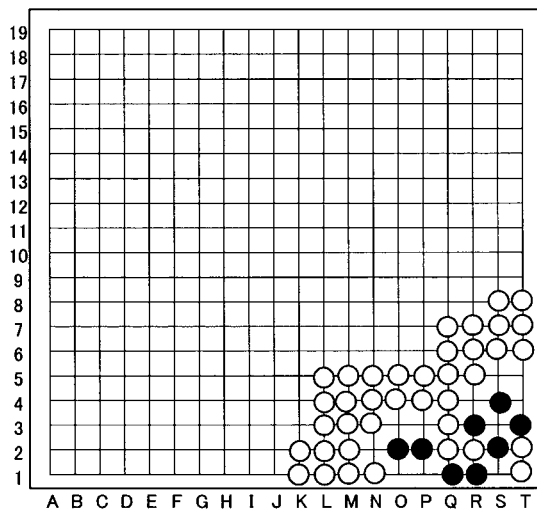


【図 3 5】

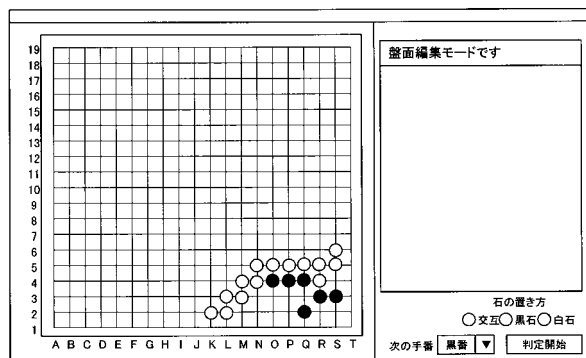


これらのE点に、外部の石が2以上含まれ、かつ判定対象の石が一つも含まれていない点かを判定。そうならば相手側の石を配置し、外部としてみなす。

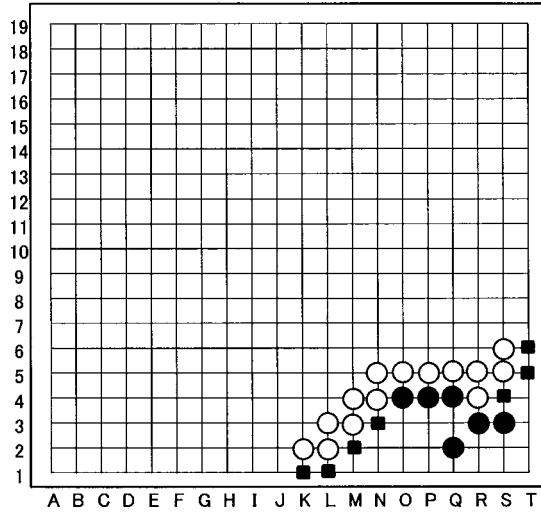
【図 3 6】



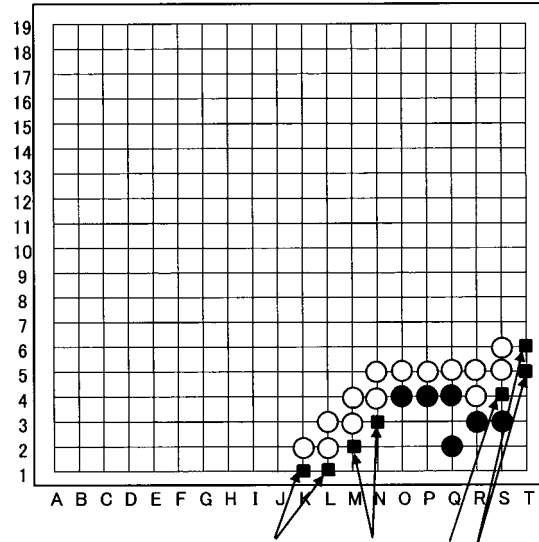
【図 3 7】



【図 38】

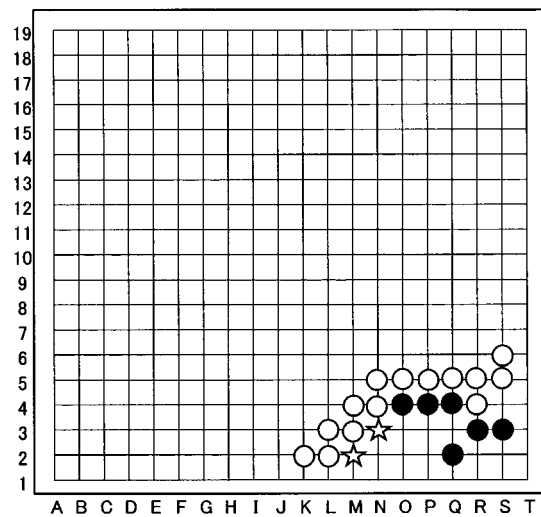


【図 39】

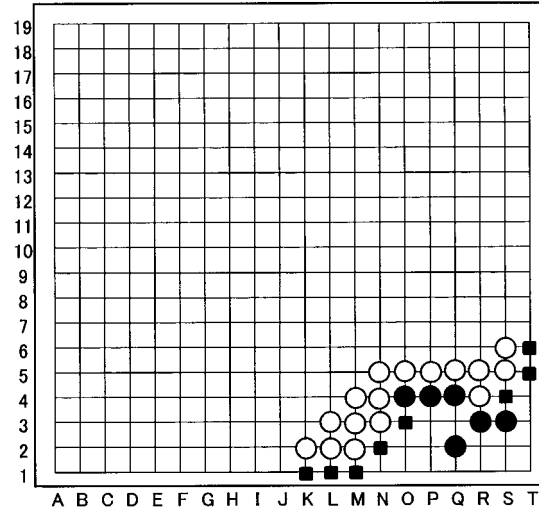


これらのE点に、外部の石が2以上含まれ、かつ判定対象の石が一つも含まれていない点かを判定。そうならば相手側の石を配置し、外部としてみなす。

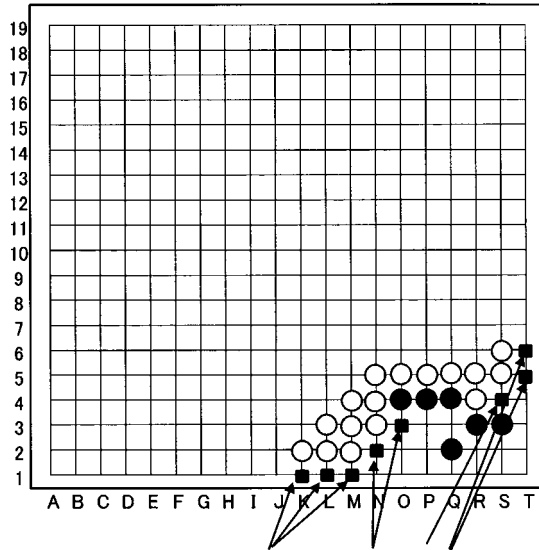
【図 40】



【図 41】

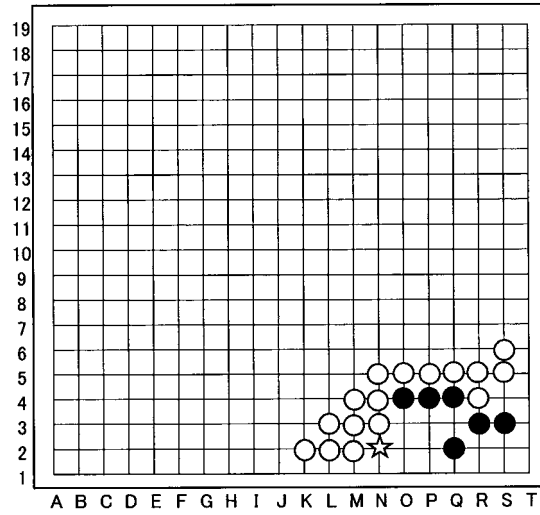


【図 4 2】

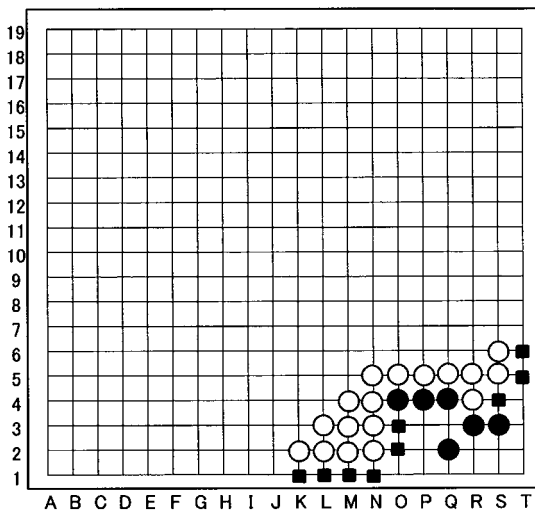


これらのE点に、外部の石が2以上含まれ、かつ判定対象の石が一つも含まれていない点かを判定。そうならば相手側の石を配置し、外部としてみなす。

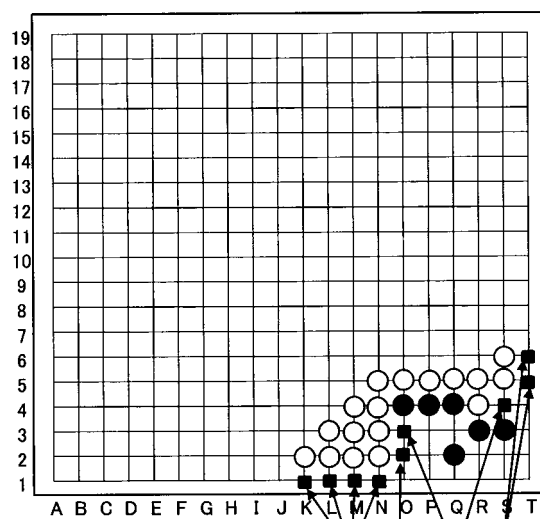
【図 4 3】



【図 4 4】

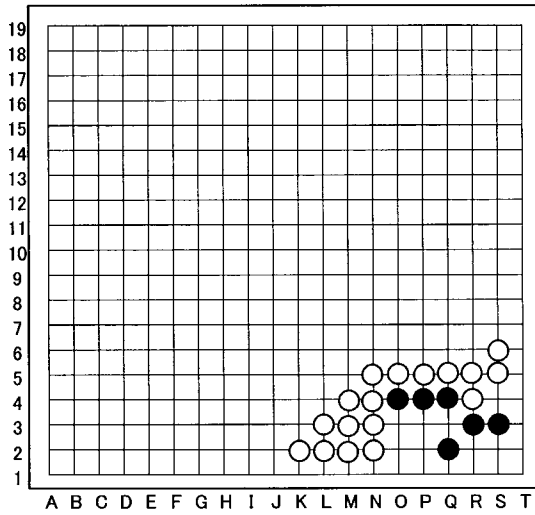


【図 4 5】

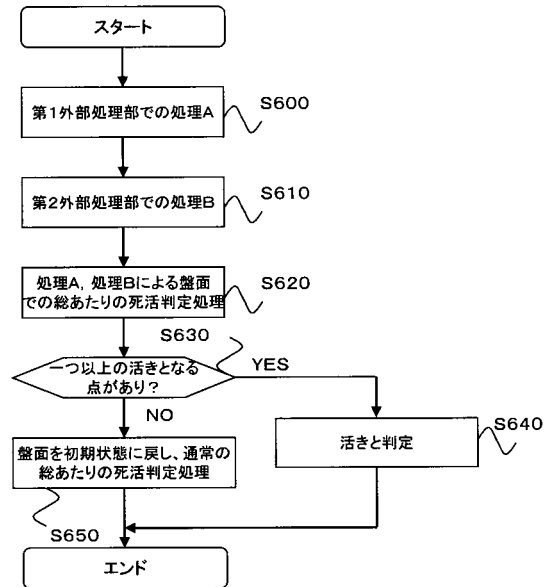


これらのE点に、外部の石が2以上含まれ、かつ判定対象の石が一つも含まれていない点かを判定。そうならば相手側の石を配置し、外部としてみなす。

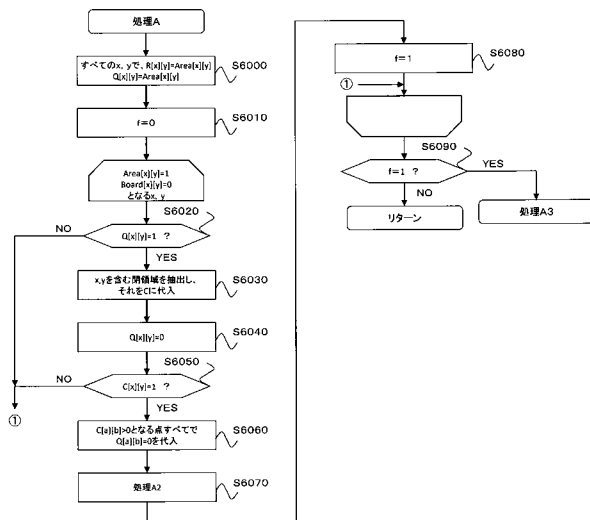
【図 46】



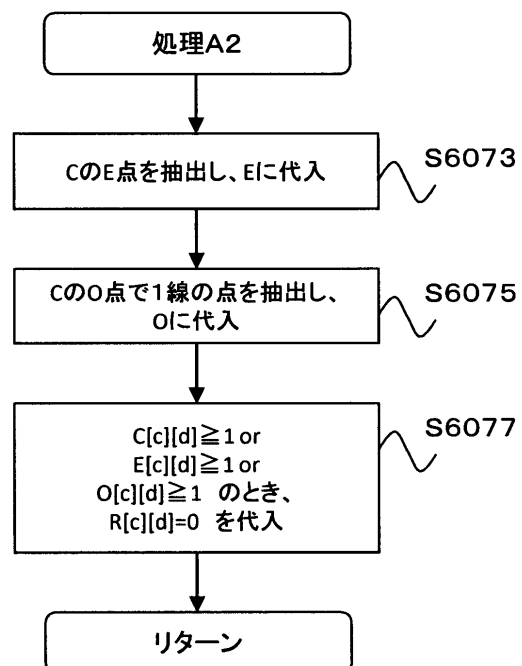
【図 47】



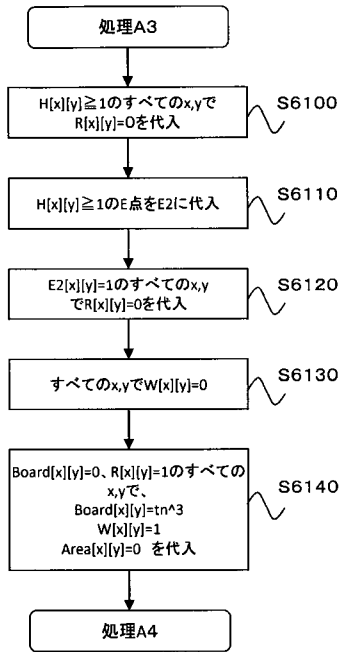
【図 48】



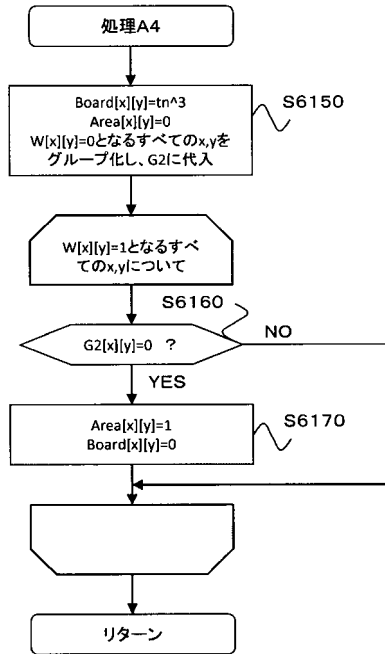
【図 49】



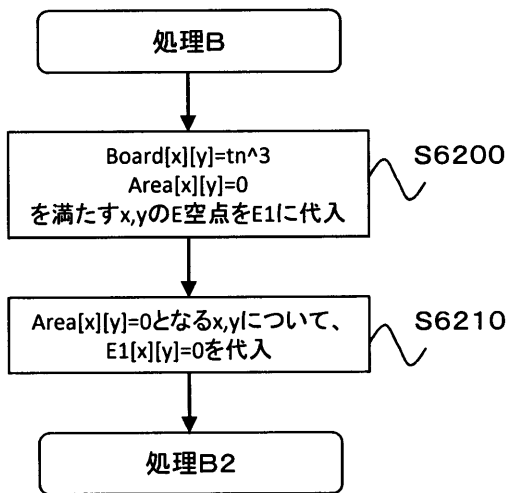
【図 50】



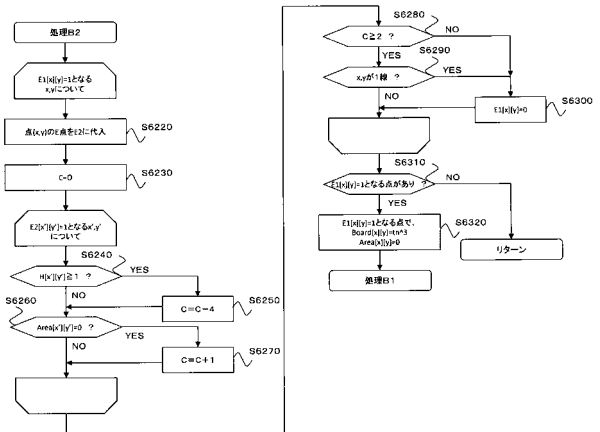
【図 51】



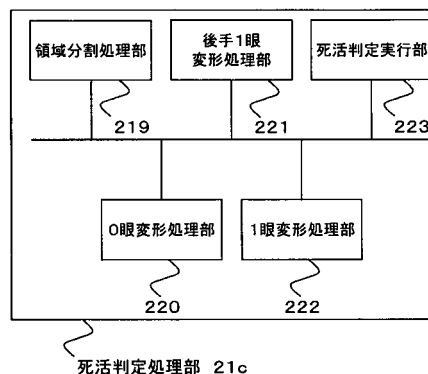
【図 52】



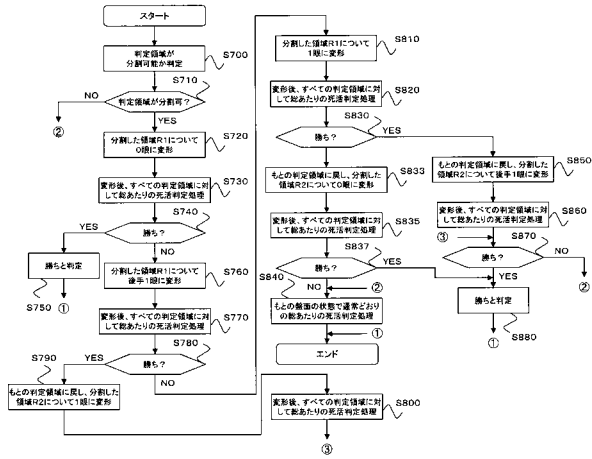
【図 53】



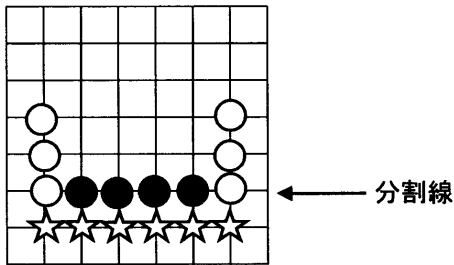
【図 54】



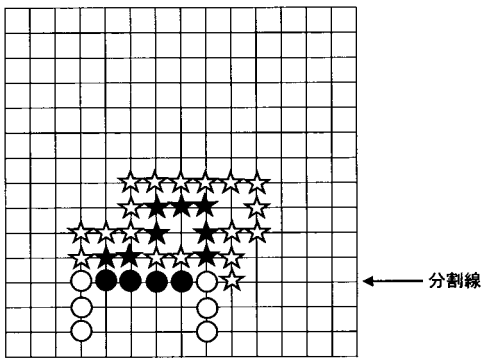
【図 55】



【図 56】

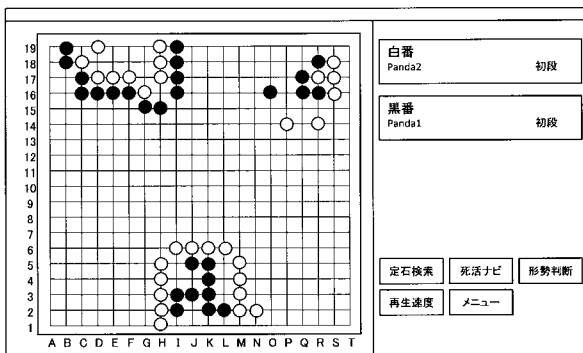


【図 58】

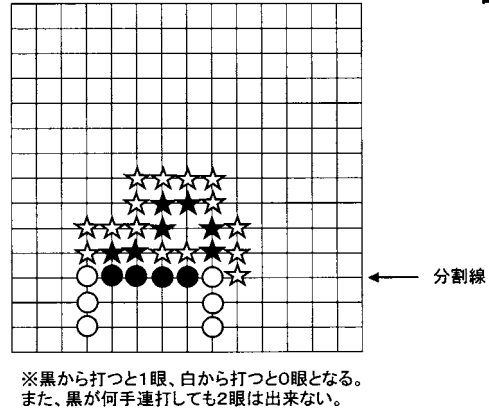


※黒から打っても白から打っても黒1眼となる。また黒が何手連打しても2眼は出来ない。

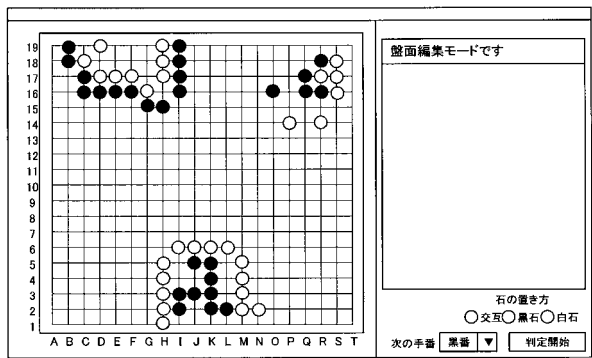
【図 59】



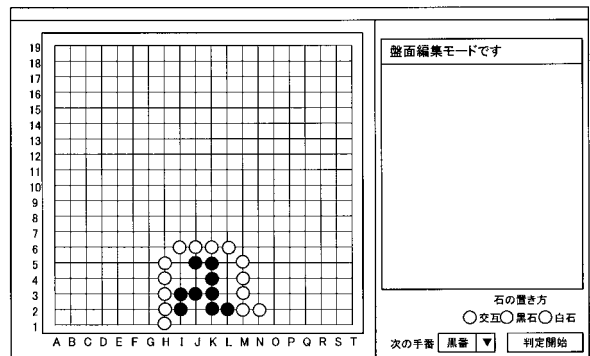
【図 57】



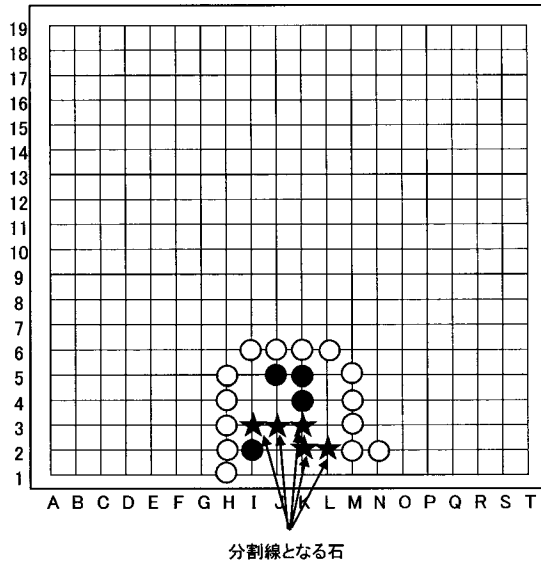
【図 60】



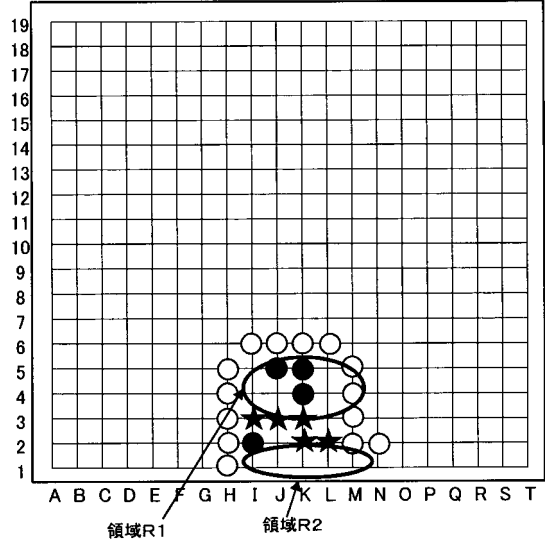
【図 61】



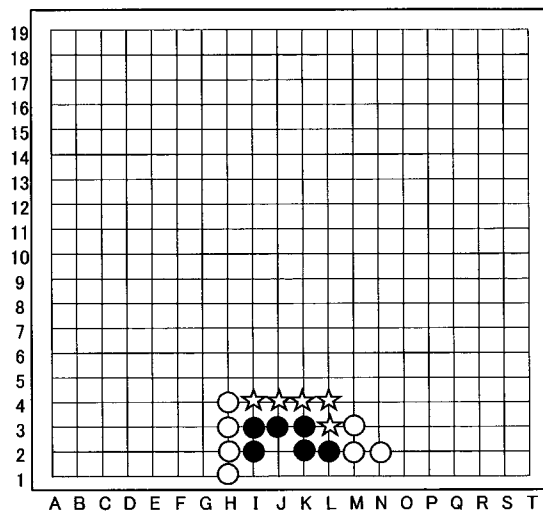
【図 6 2】



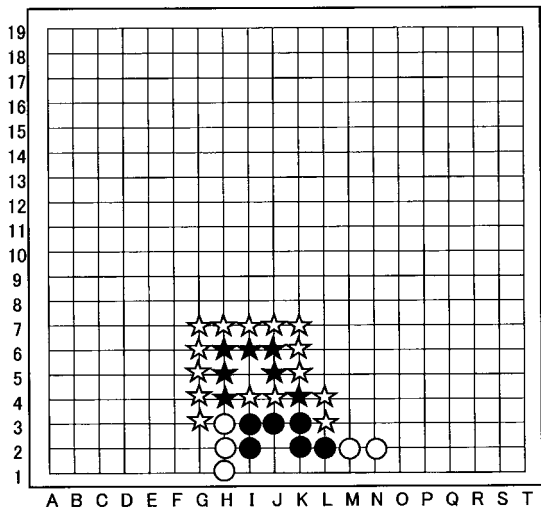
【図 6 3】



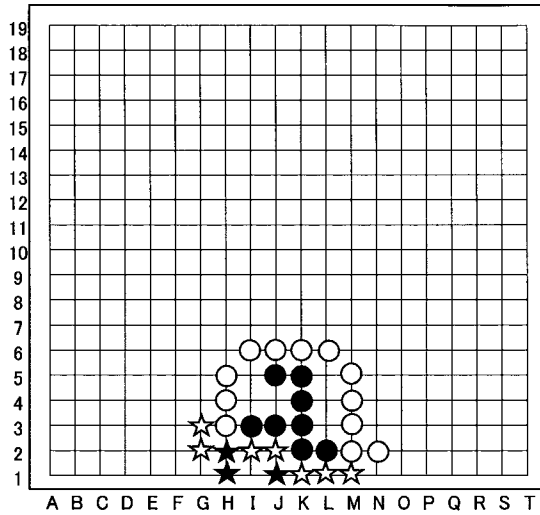
【図 6 4】



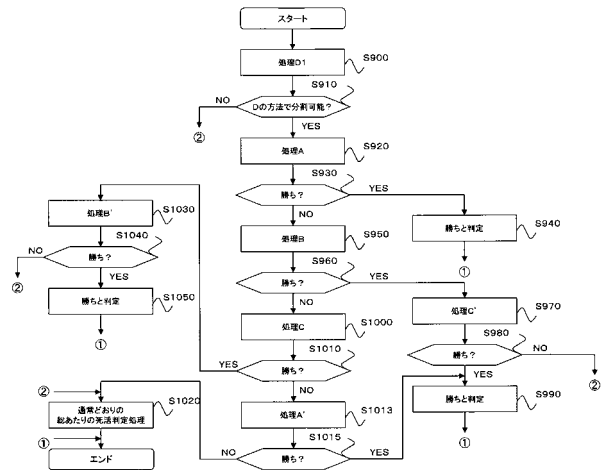
【図 6 5】



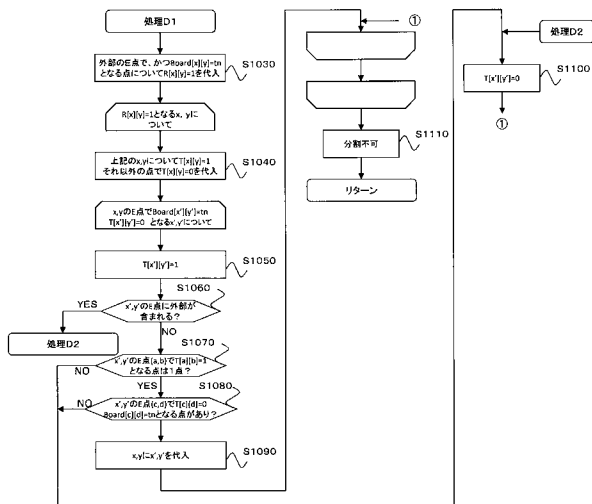
【 図 6 6 】



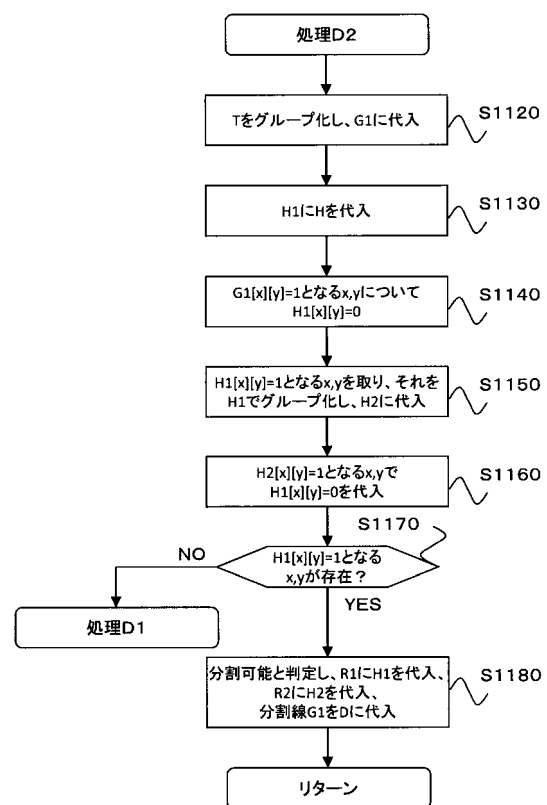
【 図 6 7 】



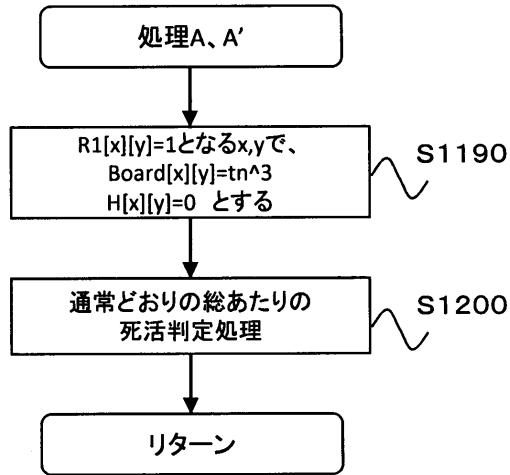
【 ㄨ 6 8 】



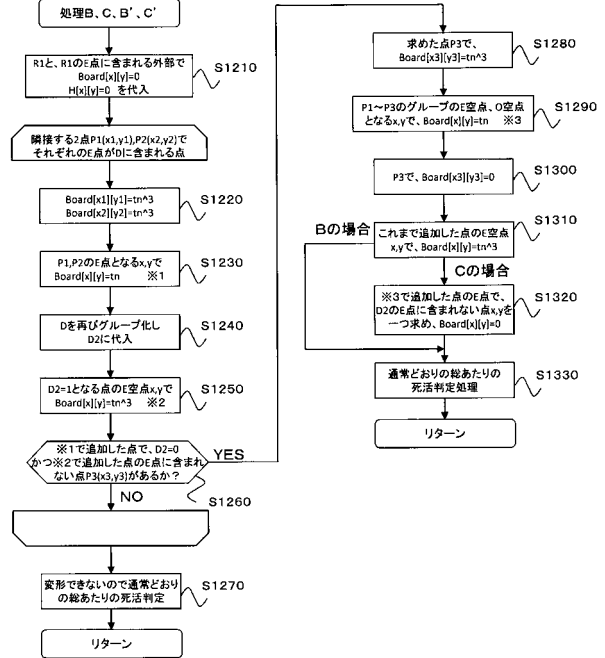
【 図 6 9 】



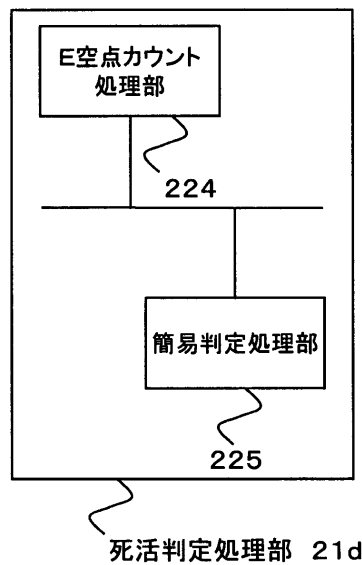
【図 70】



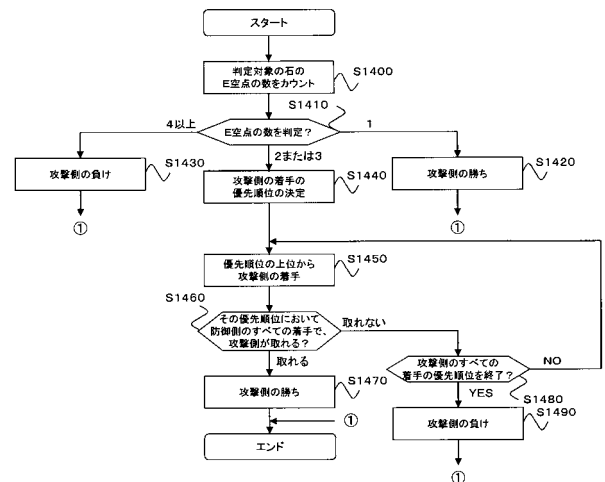
【図 71】



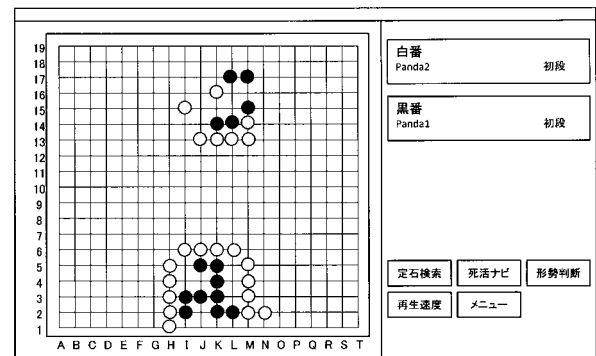
【図 72】



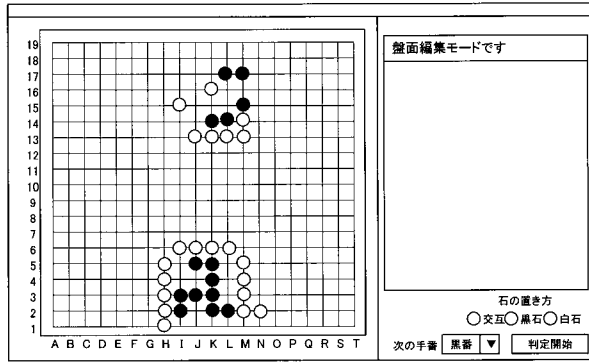
【図 73】



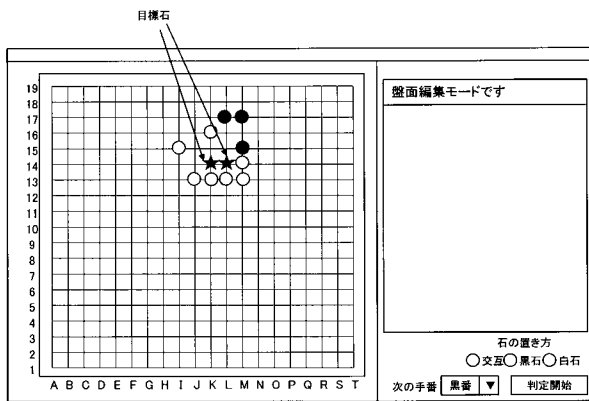
【図 74】



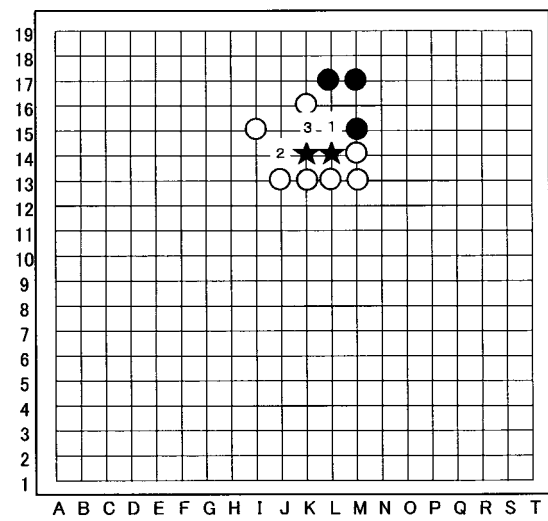
【図 75】



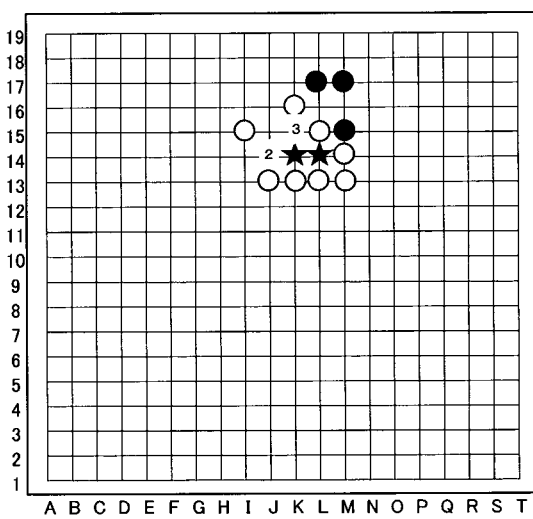
【図 76】



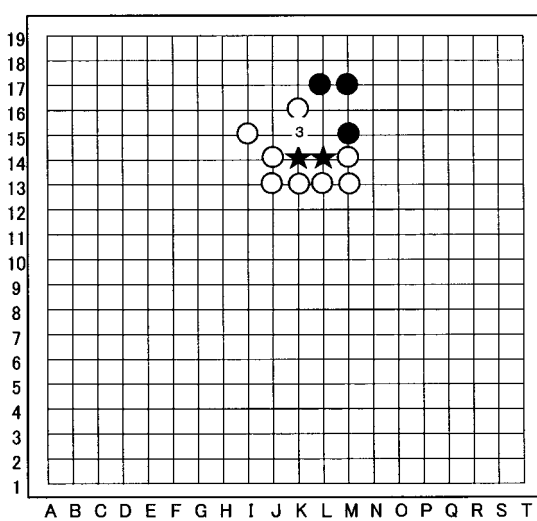
【図 77】



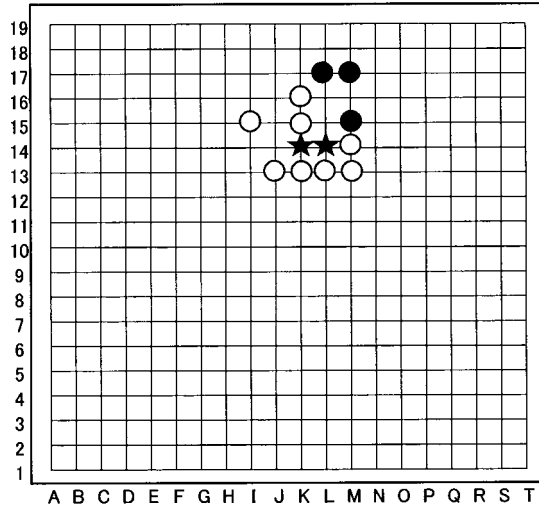
【図 78】



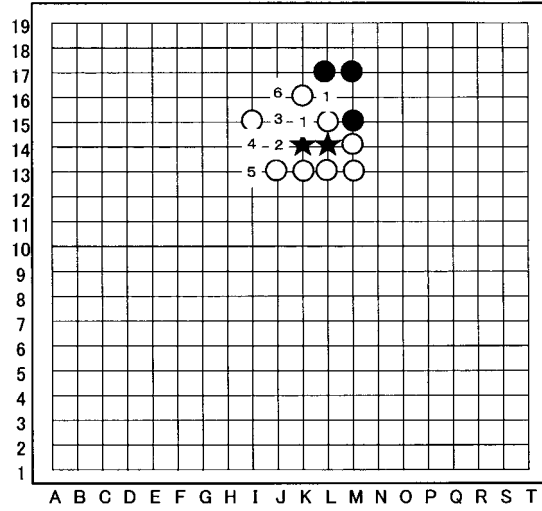
【図 79】



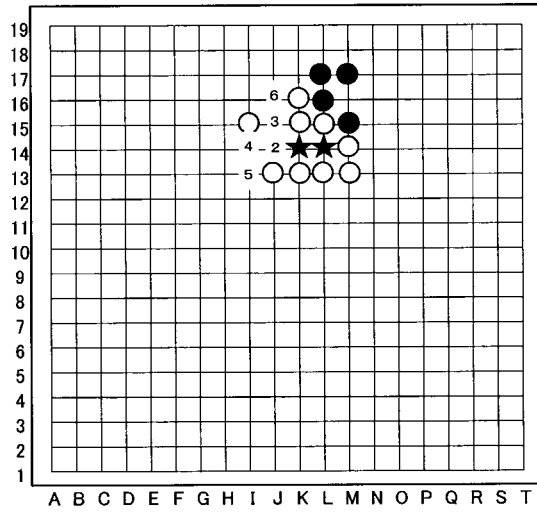
【図 80】



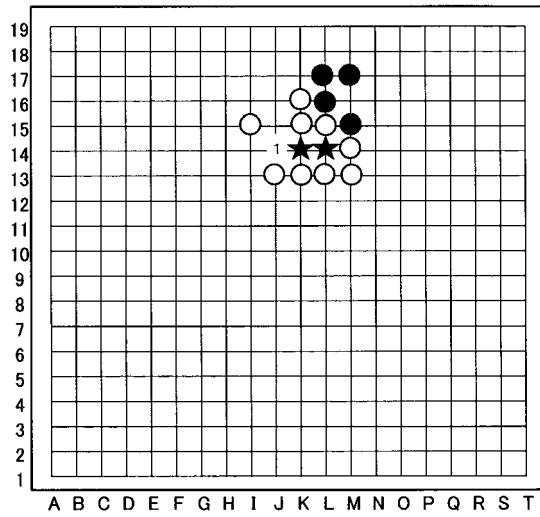
【図 81】



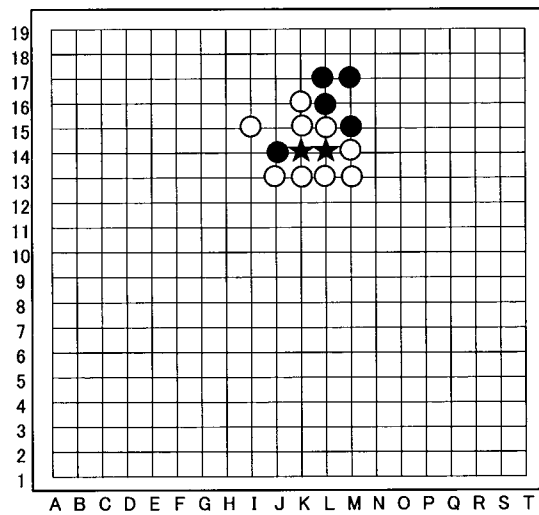
【図 8 4】



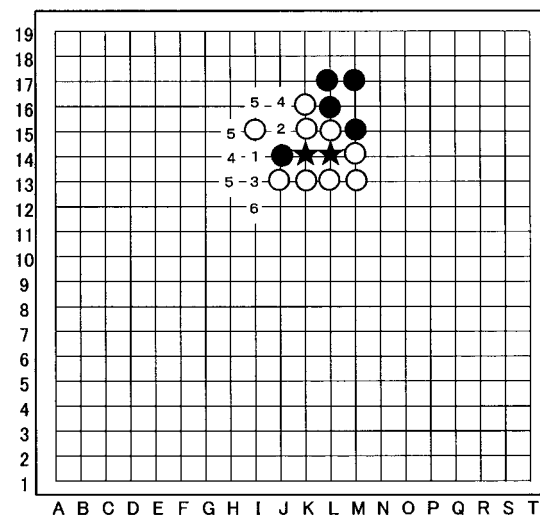
【図 8 5】



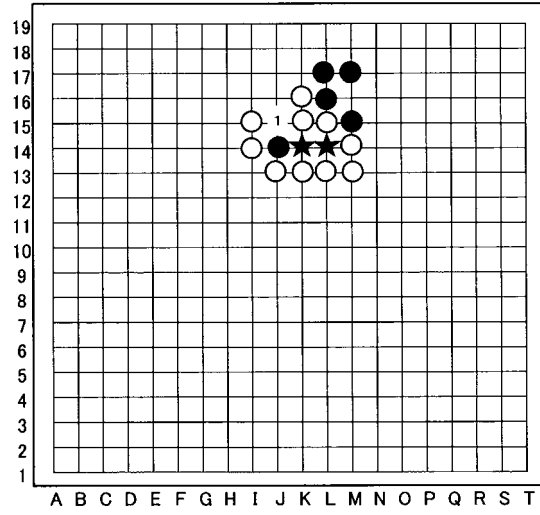
【図 8 6】



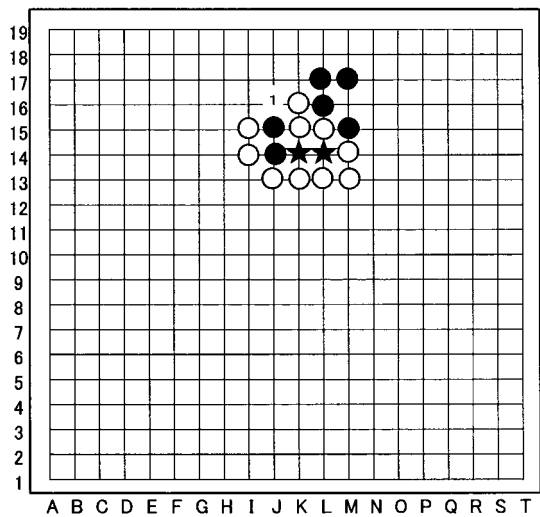
【図 8 7】



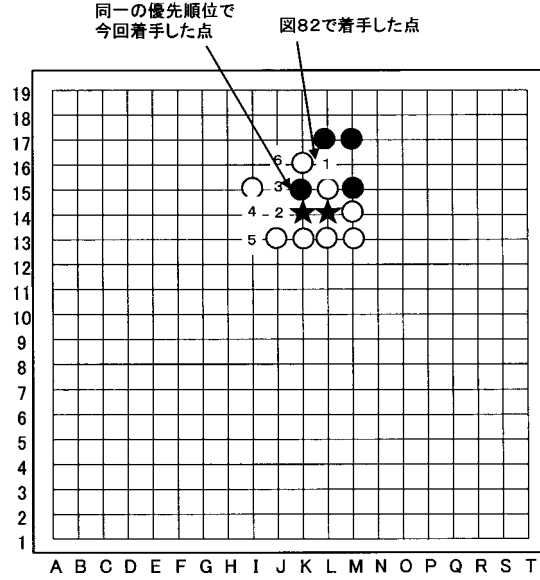
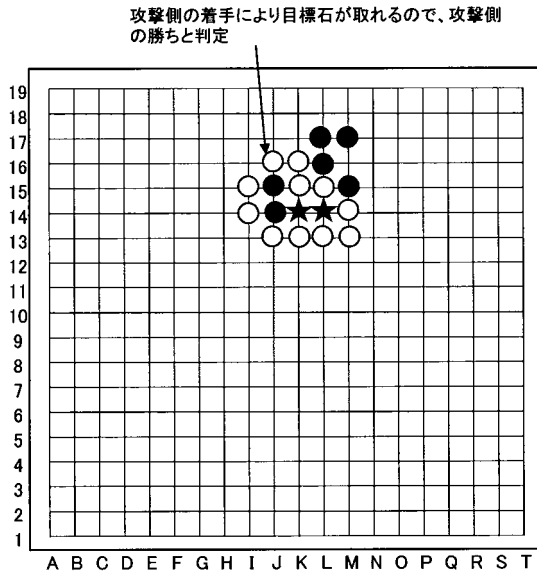
【 図 8 9 】



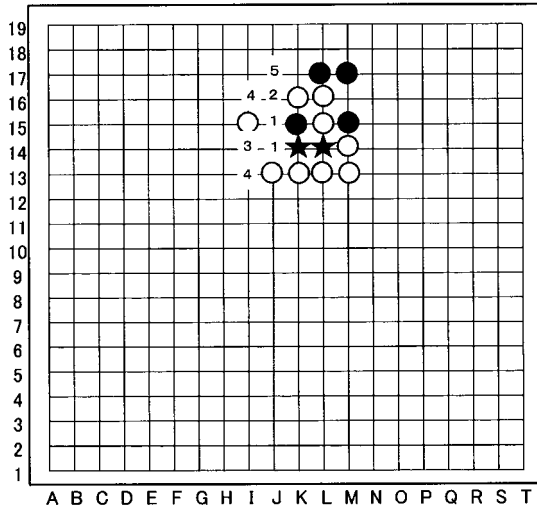
【圖 9 1】



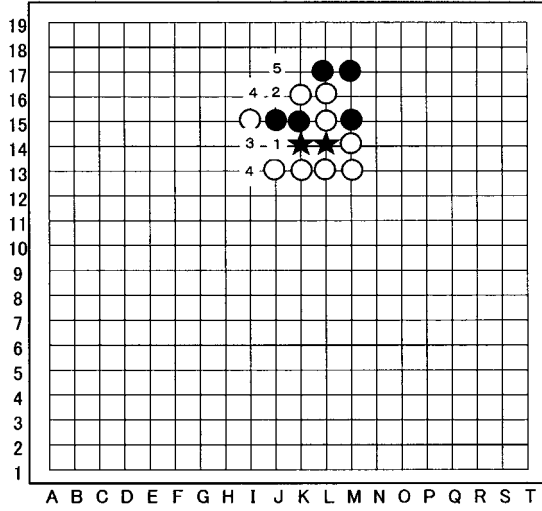
【図 9 2】



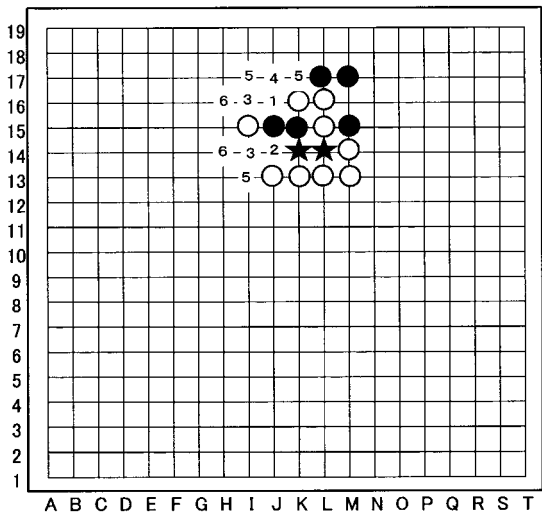
【図 96】



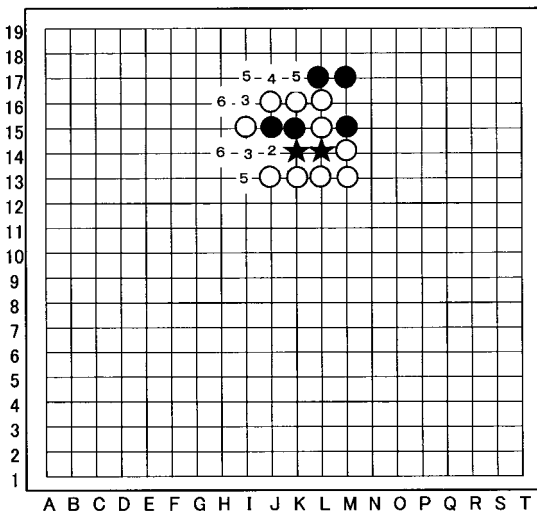
【図 97】



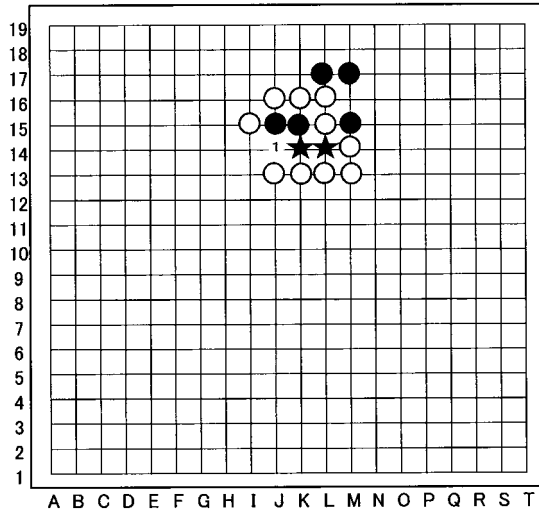
【図 98】



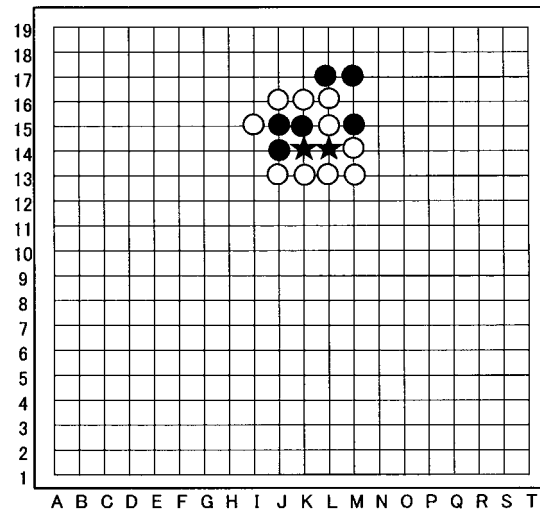
【図 99】



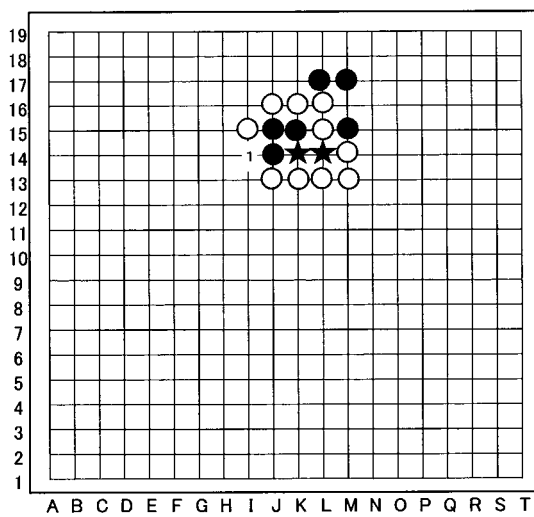
【図 100】



【図 101】

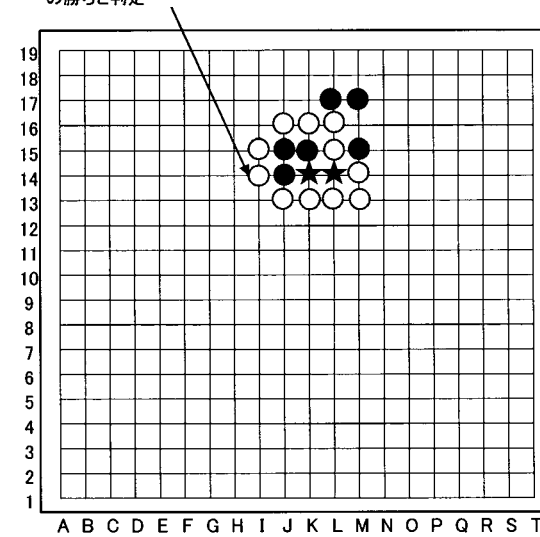


【図 102】

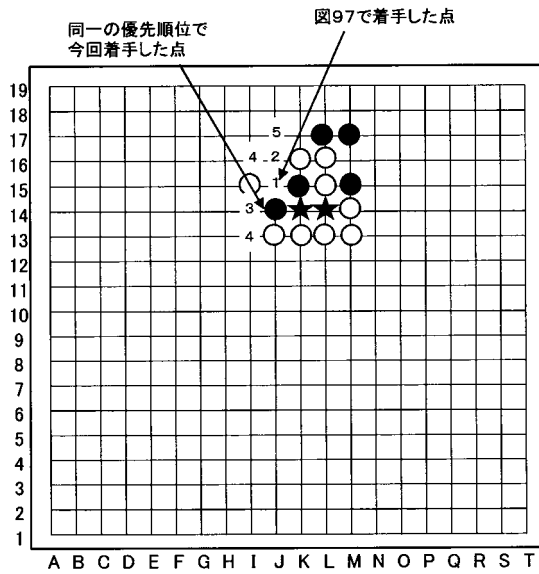


【図 103】

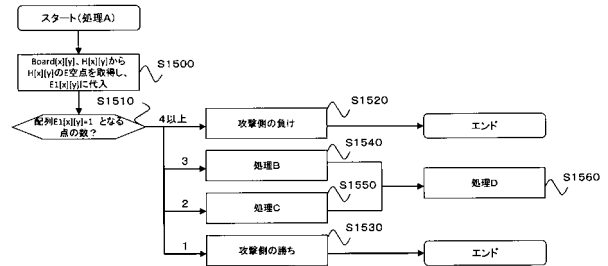
攻撃側の着手により目標石が取れるので、攻撃側の勝ちと判定



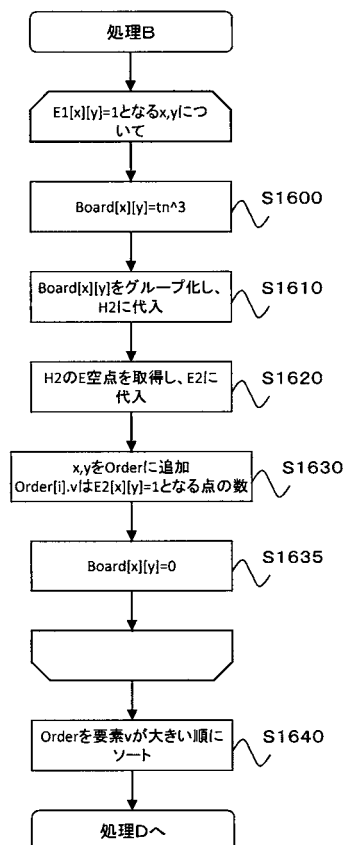
【図104】



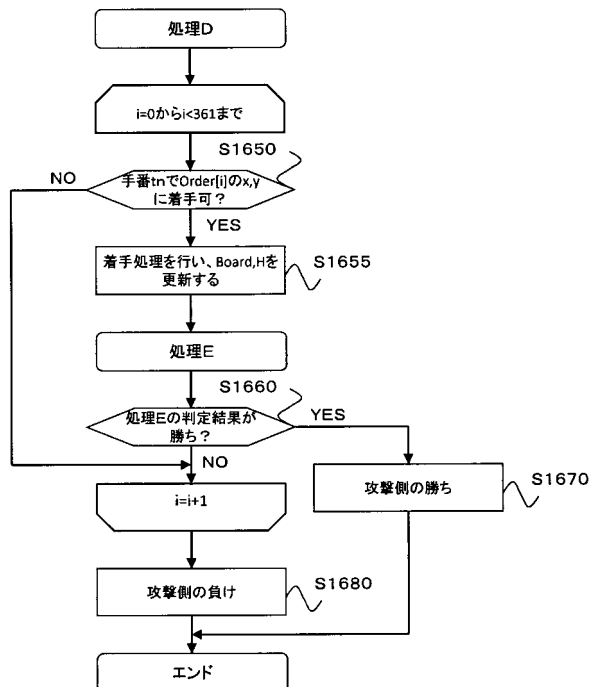
【図105】



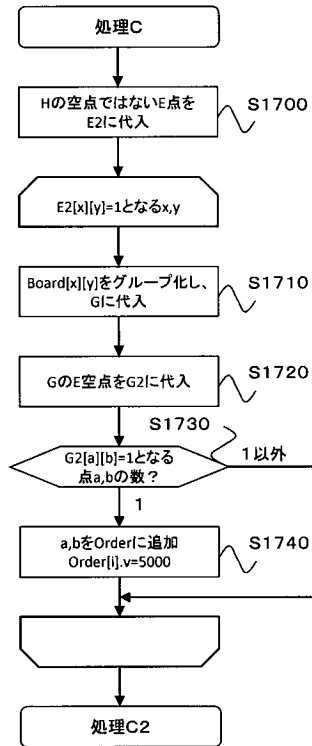
【図106】



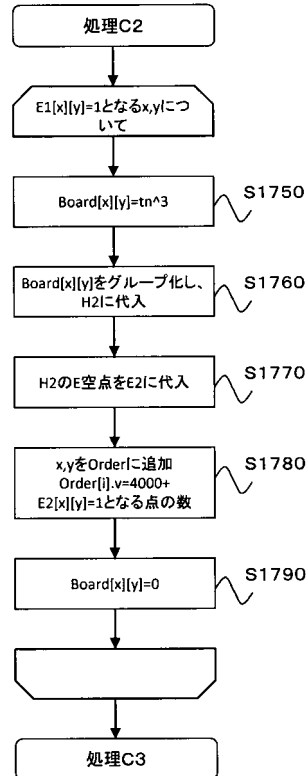
【図107】



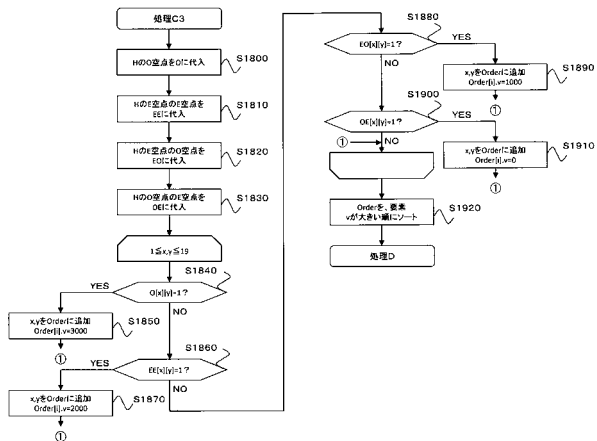
【図108】



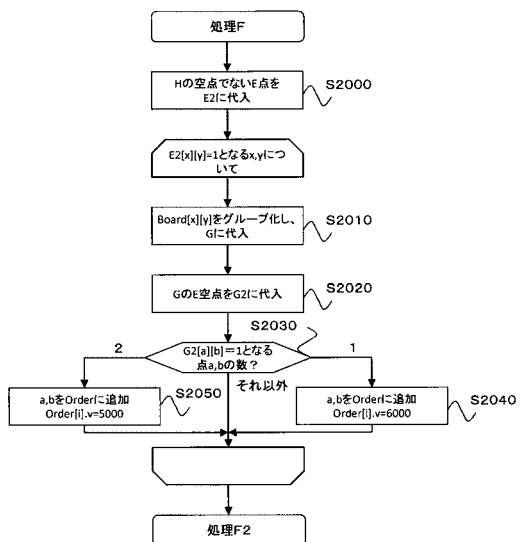
【図109】



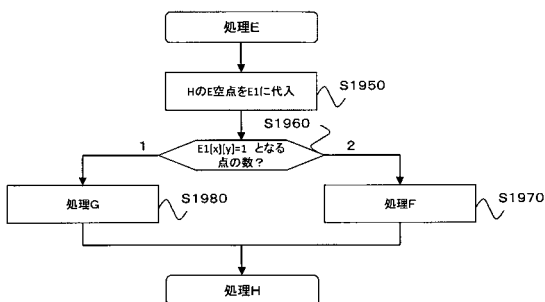
【図110】



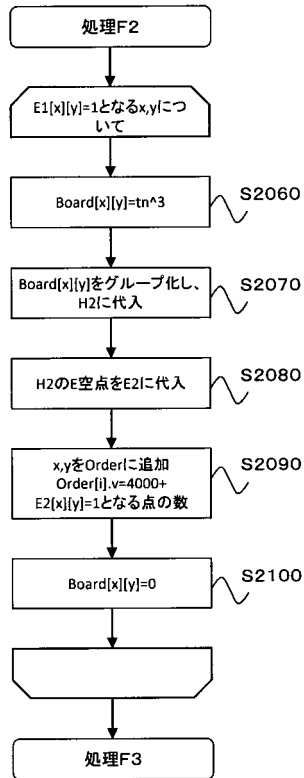
【図112】



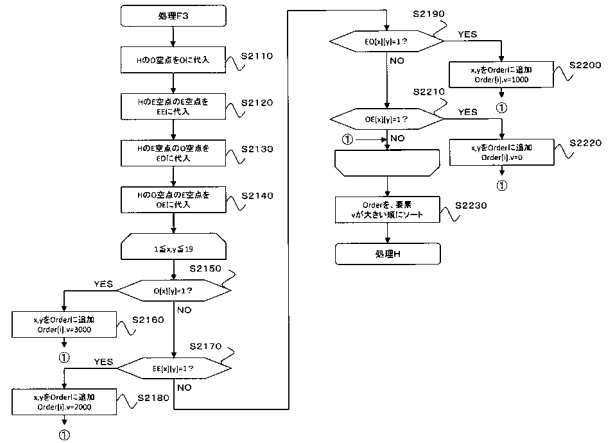
【図111】



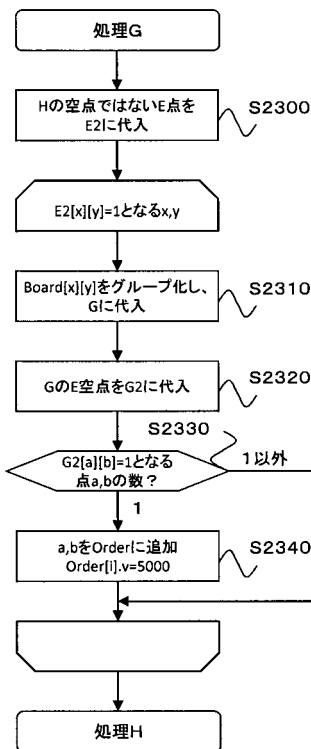
【図 1 1 3】



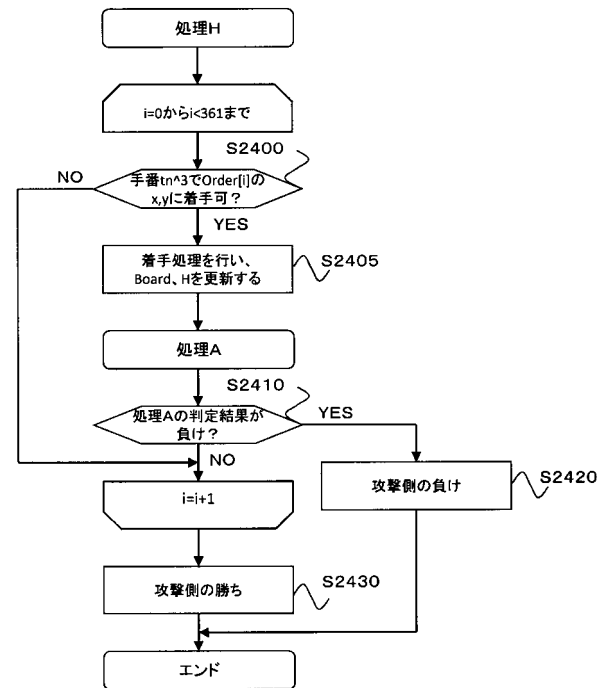
【図 1 1 4】



【図 1 1 5】



【図 1 1 6】



フロントページの続き

(72)発明者 大本 宏

東京都千代田区丸の内三丁目4番1号 株式会社パングネット内

審査官 加藤 肇

(56)参考文献 特開2002-292126(JP,A)

特開平09-075549(JP,A)

コンピュータ囲碁の入門, 日本, 共立出版株式会社, 2005年11月10日, 初版, p. 108-110, 114-115

加藤恒, 山崎勝弘, 事例を用いたコンピュータ9路盤囲碁における死活判定, 第53回(平成8年後期)全国大会講演論文集(2), 日本, 社団法人情報処理学会, 1996年9月4日, 2-145~2-146

(58)調査した分野(Int.Cl., DB名)

A63F 13/00-13/12

A63F 3/02