The Design of Gomi

Joe Hurd

joe@gilith.com

2 July 2008

Abstract

This paper presents the design of Gomi, a go playing program.

1 Introduction

This paper presents the design of Gomi,[1] a go playing program.

1.1 Notation

- $\mathbb{B}$ denotes the type of booleans $\{\mathrm{true}, \mathrm{false}\}$.
- Probabilities p.
- Probability distributions $\overline{x}$ over elements of type x.
- Points v on the board.
- Board positions ρ, including final positions τ.
- Legal moves m.
- Patterns χ: functions from ρ to $\mathbb{B}$.
- Formulas ϕ: functions from τ to $\mathbb{B}$.
- Pattern databases Π: sets of $(\chi, \phi, \overline{p})$.
- Strategies σ: functions from ρ to $\overline{m}$.

[1] Gomi is available for download from `http://www.gilith.com/software/gomi`

2 Strategy

The core algorithm of gomi evaluates a position by playing many sample games with strategy σ.

Strategy σ is the following method for selecting a move from a position ρ:

1. For each legal move m, there is a probability p_m that the position $\mathsf{move}(\rho, m)$ is winning if both players follow strategy σ.

2. Estimate the probability distribution of p_m in $[0, 1]$ from a pattern database.

3. Use these estimates to calculate the probability q_m that p_m is the maximum among all legal moves.

4. Pick a move m by sampling from probability distribution q_m.

Step 2 is the difficult one, and makes the effectiveness of strategy σ dependent on the quality of the pattern database.

3 Formulas

The key theoretical concept is

$$\mathbb{P}(\phi \mid \rho) = p$$

which means: if both players follow strategy σ starting from position ρ, the final position will satisfy formula ϕ with probability p.[2] This probability is well-defined for every position ρ and formula ϕ.

The most important property of the final position is whether it has satisfied the formula $\mathsf{BlackWins}$, which is decided by the formulas $\mathsf{isBlackTerritory}(v)$ and $\mathsf{isSeki}(v)$ for all the points v on the board.

[2]The formula ϕ having probability p can be generalized to a probability distribution over any property of final positions, such as number of seki points, but formulas are complicated enough for now.

4 Pattern Database

A pattern χ is an abbreviation for all positions that match χ, weighted by the frequency that a position appears when both players follow strategy σ. When we meet a position ρ matching χ, we want to estimate the probability $\mathbb{P}(\phi \mid \rho)$. Therefore, the pattern database stores entries of the form

$$(\chi, \phi, d)$$

where d is a probability distribution over $[0, 1]$ that estimates the random variable

$$\mathbb{P}(\phi \mid \rho) \mid \rho \text{ matches } \chi \ .$$

A useful (χ, ϕ, d) entry in the pattern database is one where χ is matched often, d is spiky, and ϕ greatly reduces the entropy of $\mathsf{BlackWins}$.

This raises two interesting questions: how do we find useful χ and ϕ pairs; and given χ and ϕ, how to calculate d? Take second question first.

4.1 Estimating Probabilities

Special case: if χ only matches one position, then we can use the frequency of ϕ being satisfied to estimate p. If χ was matched n times, and ϕ was satisfied on r of those occasions, then d can be the beta distribution with parameters $\alpha := r + 1$ and $\beta := n + 1 - r$. Abbreviate this as $B_{r,n}$.

In general, we must consider all possible splits of the formula frequency. For example, if the pattern χ being matched led to the formula ϕ being satisfied with probability $1/2$, then this might mean either: that all positions that match χ satisfy ϕ with probability $1/2$; or half the positions that match χ satisfy ϕ with probability 1, and the other half satisfy ϕ with probability 0.

Let

$$q = \mathbb{P}(\phi \mid \chi) \ ,$$

then a conservative estimation of the probability

$$\mathbb{P}(\phi \mid \rho) \mid \rho \text{ matches } \chi$$

is

$$
\begin{aligned}
d_q(p) &= \text{The maximum proportion of positions that can have probability } p \\
&= \max\{x \mid \exists p' \in [0,1].\ p * x + p' * (1-x) = q\} \\
&= \max\{x \mid \exists p' \in [0,1].\ x * (p - p') + p' = q\} \\
&= \mathsf{if}\ p > q\ \mathsf{then}\ q/p\ \mathsf{else}\ (q-1)/(p-1) \\
&= \mathsf{if}\ p > q\ \mathsf{then}\ q/p\ \mathsf{else}\ (1-q)/(1-p)
\end{aligned}
$$

The probability q is unknown, but we can estimate it using the beta distribution $B_{r,n}$, to give the following estimate:

$$d(p) = \int_{q \in B_{r,n}} d_q(p)\ .$$

This is (expected to be) pretty spiky when q is close to 0 or 1, but is not much help otherwise. To further refine the estimation, we can keep track of which patterns are reachable in one move from χ.

Acknowledgements

Alistair Turnbull read an early draft of this design document.