GoNN - Incorporating a Neural Network into the Game of Go

C. Fellows, Y. Malitsky, G. Wojtaszczyk

Go Seminar, University of Alberta; Presented by M. Enzenberger

April 18, 2006

Contents

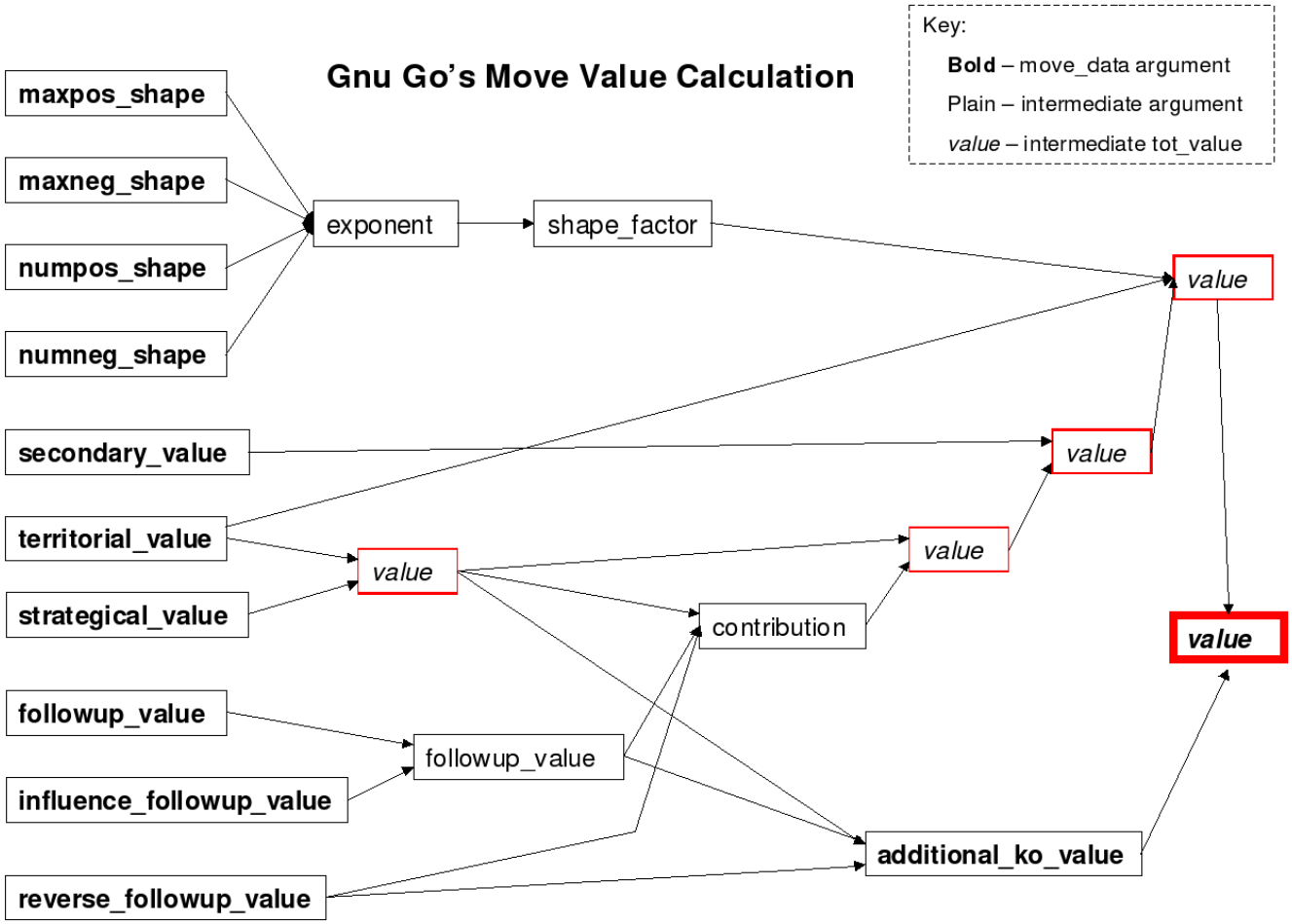
Task Definition

- Improve GNU Go's move evaluation
- GNU Go's characteristic values as input
- Replace combination of values using machine learning

Description of GNU Go's Values

Value	Description
Territorial Value	The total amount of territory the player is expected to gain by playing this move
Strategical Value	Measure of the effect the move has on the safety of all chains of stones on the board
Max Pos Shape	How good is the best shape formed by this move as determined by the pattern matching code
Max Neg Shape	How bad is the worst shape formed by this move as determined by the pattern matching code
Num Pos Shape	Number of good shapes formed by this move as determined by the pattern matching code
Num Neg Shape	Number of bad shapes formed by this move as determined by the pattern matching code
Follow-up Value	Value which may accrue if the player gets two moves in a row in local area
Influence Follow-up Value	How much territory can be gained if the player gets two consecutive moves in local area
Reverse Follow-up Value	The value the opposing player gains if allowed two consecutive moves in a local area
Secondary Value	Given for move reasons which by themselves are not sufficient to play the move
Min Value	The minimum value this move can get as determined by the pattern matching code
Max Value	The maximum value this move can get as determined by the pattern matching code

GNU Go's Move Value Calculation



Algorithm Definition

Correct move values are not available
$\rightarrow$ Maximize difference between correct move and all others

$$\text{Maximize } \epsilon$$

$$\text{Subject to: } \sum (\mathrm{I}_{\text{expert}} - \mathrm{I}_{\text{other}}) W \geq \epsilon$$

$$\epsilon \leq \text{constant}$$

[*Don't understand this notation*]

- Solve with Simplex Algorithm
  [*How does the above map to the simplex algorithm?*]
- Solve with Support Vector Machine

Simplex Algorithm

Maximize

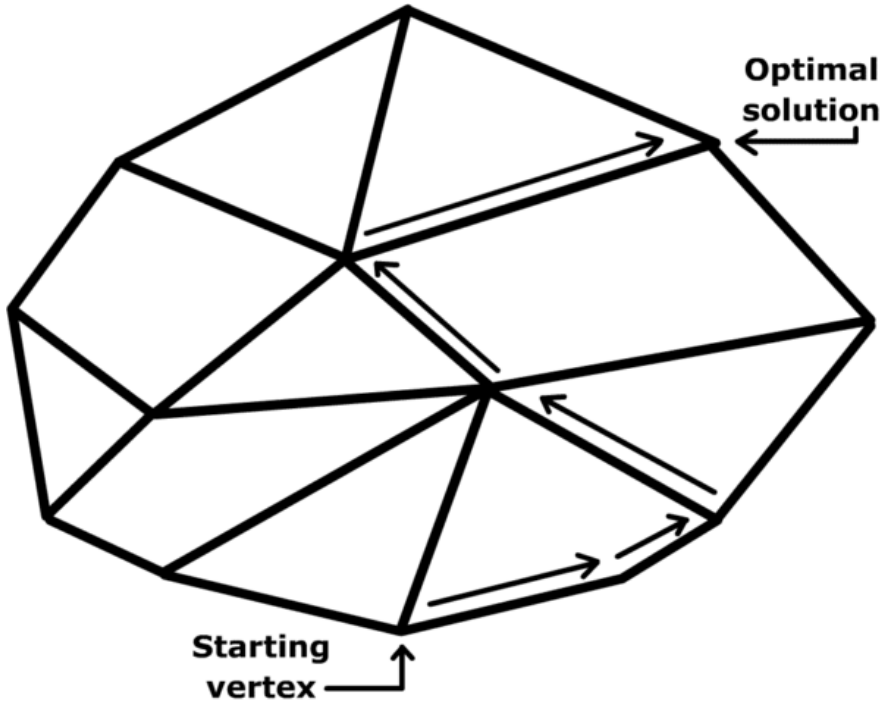
$$\mathbf{c}^T\mathbf{x}$$

subject to

$$\mathbf{Ax} \leq \mathbf{b}, \quad \mathbf{x} \geq 0$$

- Linear programming problem
- Simplex Algorithm: numerical solution method
- Uses a simplex
  (generalization of triangle to N dimensions)

Simplex Algorithm (cont'd)



Move from vertex to vertex along edges to find optimum solution

Support Vector Machine

Classification of data points $\mathbf{x}_i$, $c \in \{-1, +1\}$

$$\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \ldots, (\mathbf{x}_n, c_n)\}$$

Dividing hyperplane

$$\mathbf{w} \cdot \mathbf{x} - b = 0$$

Maximum margin (maximum distance to data points); maximize distance of parallel hyperplanes, so that no data points are between them

$$\mathbf{w} \cdot \mathbf{x} - b = +1 \quad (1)$$

$$\mathbf{w} \cdot \mathbf{x} - b = -1 \quad (2)$$

Support Vector Machine (cont'd)

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x}_i - b &\geq +1 \qquad \text{or} \\ \mathbf{w} \cdot \mathbf{x}_i - b &\leq -1 \end{aligned}$$

Can be rewritten as

$$c_i(\mathbf{w} \cdot \mathbf{x} - b) \geq 1 \qquad 1 \leq i \leq n \tag{3}$$

Minimize $|\mathbf{w}|$ subject to constraint (3)

Support Vector Machine (cont'd)

- SVM is a Quadratic Programming optimization problem
- Non-linear classification with the kernel trick
  (non-linear transformation of the feature space)
- Soft margin method, if no hyperplane can split the data
- Can be extended to solve regression problems with real-valued output:
  $\longrightarrow$ value is distance to hyperplane

Training

- Use 800 professional games
- Only positions where GNU Go generates the correct move [*why?*]

Computer Go Test Collection

- GNU Go: 71%
- Simplex: 62%
- SVM: 67.5%

Resulting Weights

	Territorial Value	Strategic Value	Max Pos Shape	Max Neg Shape	Num Pos Shape	Num Neg Shape	Follow-up Value	Influence Follow-up Value	Reverse Follow-up Value	Secondary Value
Simplex	2441.78	2856.29	0	0	7885.43	0	441.295	1781.21	8644.67	5534.93
SVM	0	0.00901	0.00544	0.016059	-0.0187	0.01269	-0.0099	0.00011	-0.0095	0.004103

Share nothing in common

Recent Work: AAAI Poster

- Only SVM
- Computer Go Test Collection as training set
- Only tests for a correct move
- Only positions solved by GNU Go [*why?*]

- SVM finds perfect seperation
- Similar performance to GNU Go on a test set with professional games
- Add unsolved problems

SVM Values

	GnuGo	SVM	Cut / Connect		Escape Capture	
Number of Games	187	187	50	59	15	21
Territorial Value	1	14.47	15.1	20.0	0.43	3.97
Strategical Value	1	14.74	14.2	20.3	0.02	-2.8
Max Pos Shape	*	9.43	4.05	44.5	0.75	-2.5
Max Neg Shape	*	-8.93	-8.3	3.40	-0.3	0
Num Pos Shape	*	3.75	2.58	-33	0.42	21.8
Num Neg Shape	*	-4.46	-4.2	-60	-0.2	0
Follow-up Value	$	1.85	0	5.62	0	2.03
Influence Follow-up	$	7.35	4.79	6.54	-0.3	2.16
Reverse Follow-up	$	9.46	0	0	0.01	5.03
Secondary Value	0.05	0	0	0	0	40.3
Num Connected	*	2.74	1.67	10.1	0.02	8.02
Min Value	#	5.22	0.22	21.3	0.01	-0.1
Max Value	#	0.03	0.01	0.10	0	0