About SgStrategy: a simple implementation of strategies (and how we can use it)

Martin Müller

Go Seminar, Feb. 2007

Contents

- What do I mean by “strategy”
- Examples
- Applications
- Preliminary implementation
- Next steps

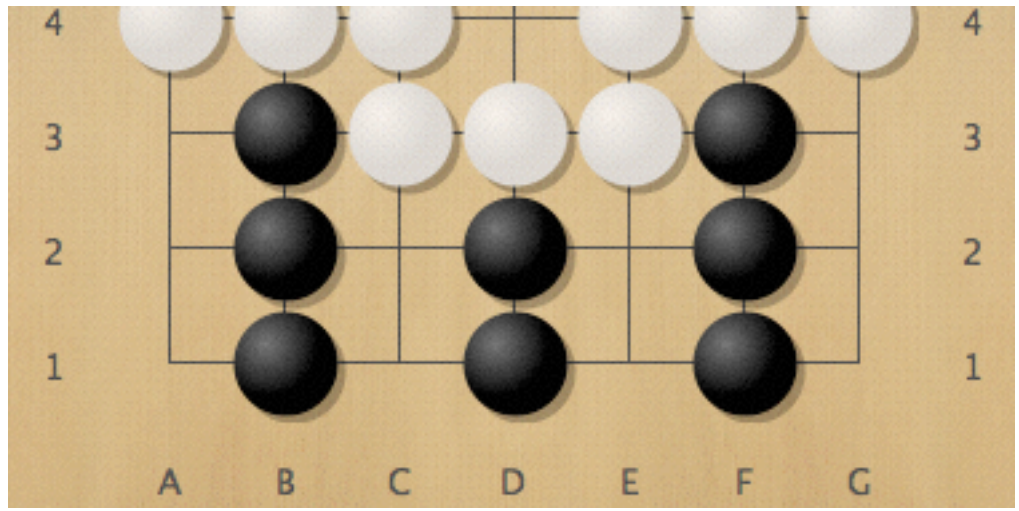
What do I mean by "strategy"

- Vague term: e.g. strategy vs tactics
- Here: a proven way to achieve a goal
  - Example: proof tree
  - Example: miai strategy
- Is there a better term?

Motivation

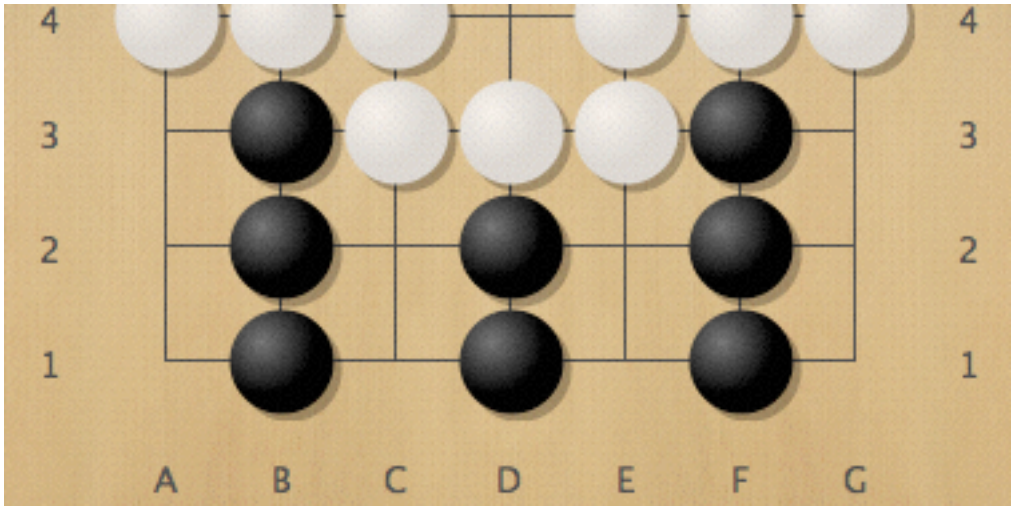
- Much Go knowledge represented implicitly using heuristic rules
  - only tested while constructing connections, dividers, groups, zones,...
  - no chance to check accuracy by other means
  - cannot handle dependencies etc.
- Want to find, use direct representation

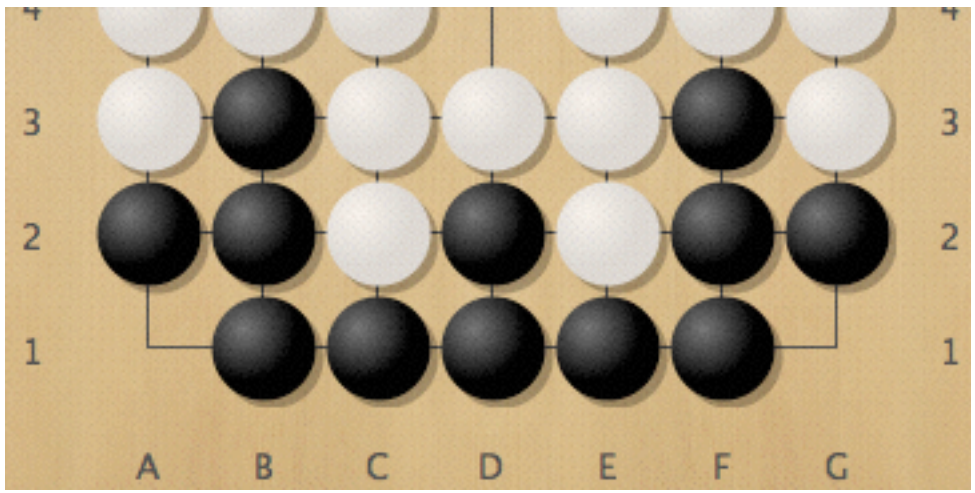
Example



- Black is alive, even if White plays first
- What is Black's strategy for living?
- How can we represent that strategy?

Example - Proof Tree





- For each W move, give one B answer, such that all lines end in a B win
- Example:
  1. W A3 - 2. B A2 -
  3. W C2 - 4. B C1 -
  5. W E2 - 6. B E1 -
  7. W G3 - 8. B G2 -
  alive by static evaluation

Proof Tree(2)

- Branching factor at start: 10
- much repetition in search
- Not the way we analyze such problems - we use subgoals eye and connection

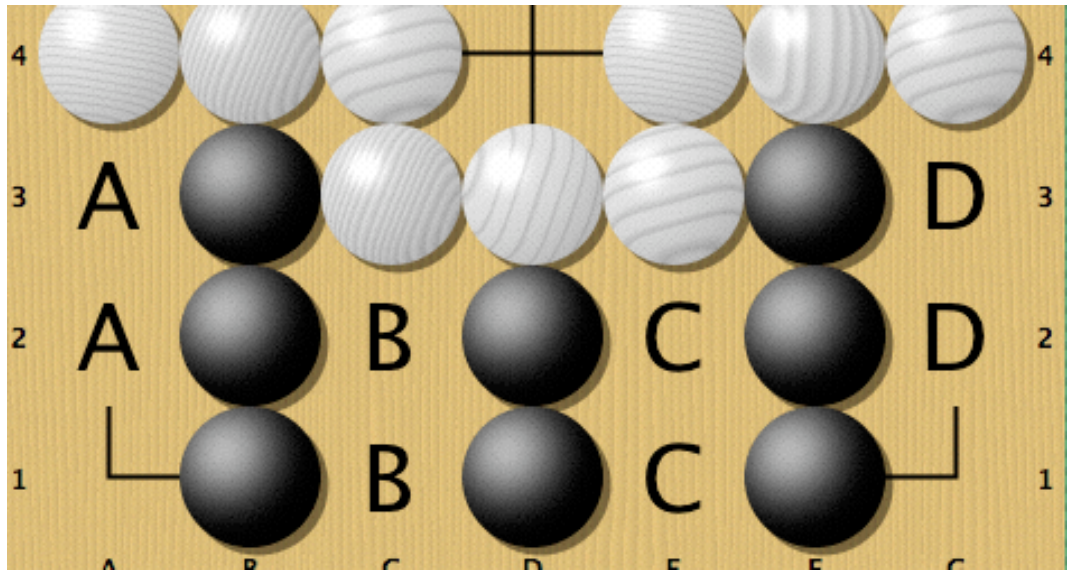
Composite strategies

- Logic of goals, subgoals
- Live(B1) = EyeAt(A1) and EyeAt(G1) and Connect(B1,F1)
- Connect(B1,F1) = Connect(B1,D1) and Connect(D1,F1)

Miai strategies

- Miai: play(A) or play(B)
- EyeAt(A1): play(A3) or play(A2)
- Connect(B1,D1): play(C2) or play(C1)
- Connect(D1,F1): play(E2) or play(E1)
- EyeAt(G1): play(G3) or play(G2)

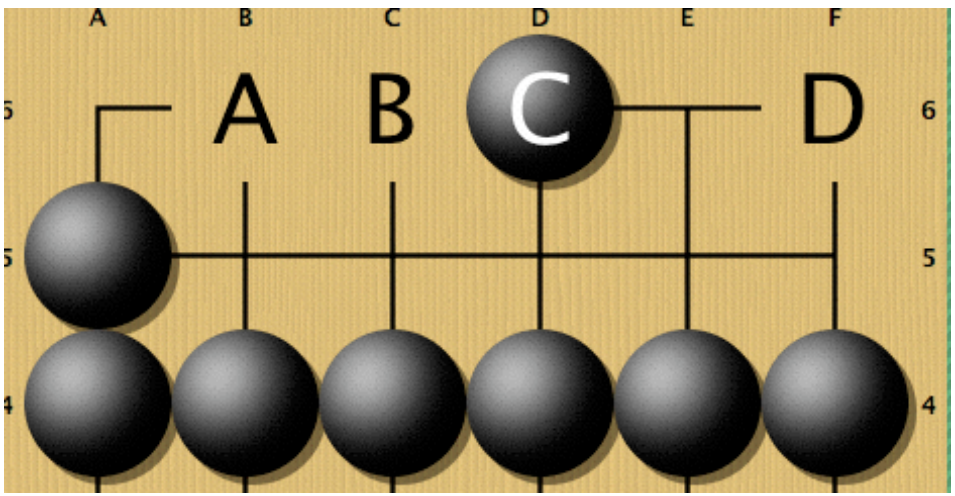
Miai Map

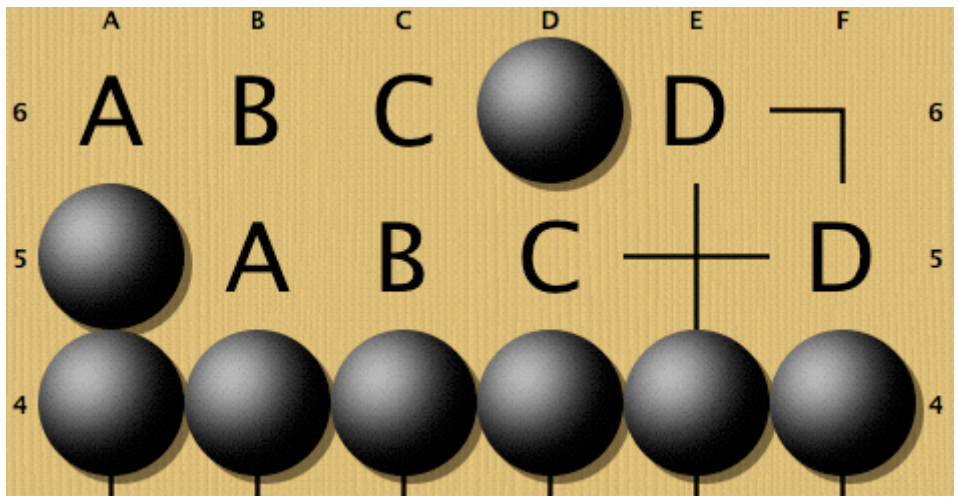


- Compact, fast representation of set of miai strategies
- Board array, for each point contains miai point

An Application

- Static Safety Solver: Miai strategy to access interior points
  - Guarantees one sure liberty





Dynamic Decomposition Search

- keep barrier intact
- similar idea, ad-hoc implementation

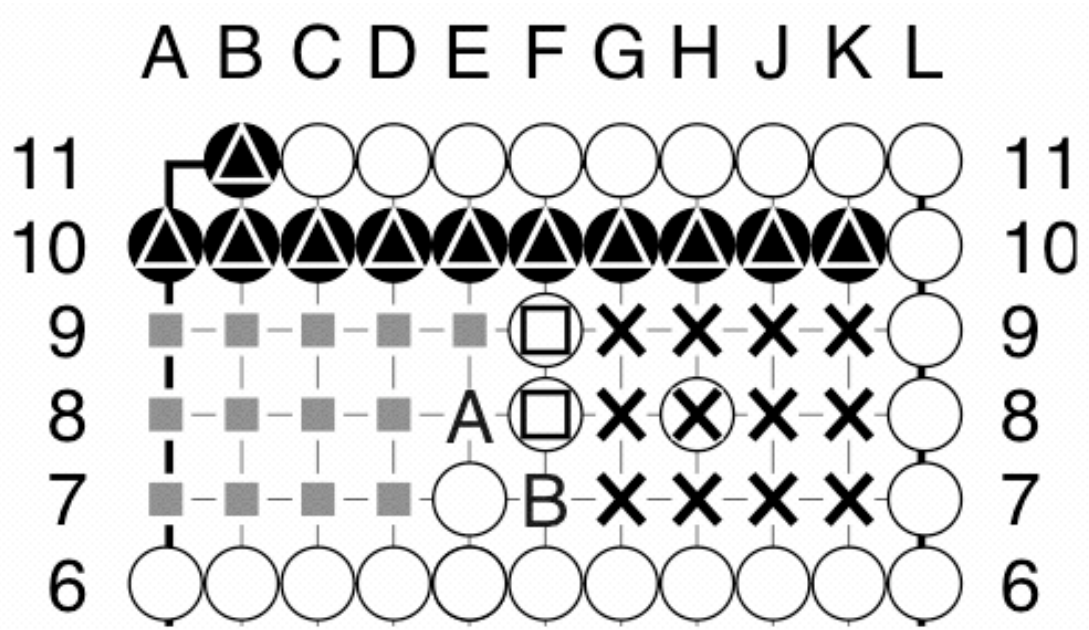


Figure 10: Relaxed decomposition.

Open Boundary Safety Solver

- keep boundary intact
- similar idea, ad-hoc implementation

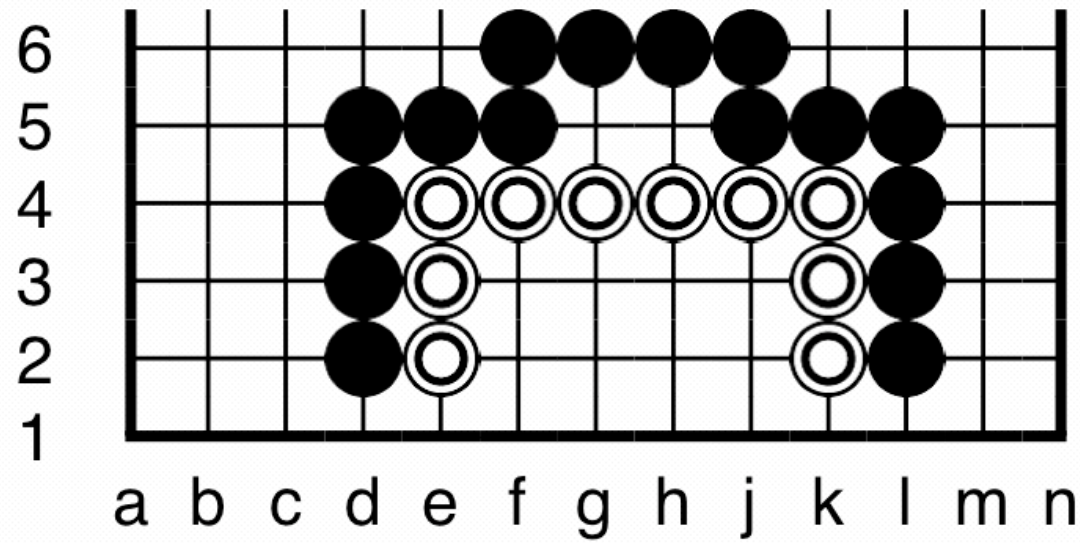


Fig. 6. external move generation for an open boundary area

Goal

- general framework
- support for using it with searches, when building knowledge-based data structures

Strategies in Play

- Find them
- Store them
- Track them
- Use them

Track and Use

- status of strategy: achieved, threatened, (unknown), failed
- both players' moves change status
- threatened - move to save strategy is known

Preliminary implementation

- SgStrategy
- SgMiaiStrategy
- SgMiaiMap

Possible Extensions

- Combine strategies, check independence
- Get proof trees from searches
- Implicit strategy - recompute on demand
- 1-level shallow partial proof trees, with re-search

Possible applications

- Use throughout Explorer
- Use in UCT player
- Relation to Hex - virtual connections?

In Explorer

- Represent low level: connections, dividers
- Build complex strategies: defend zone, group
- Represent goal-directed search outcomes
- Find and solve conflicts
- Use for
  - finding threats, double threats
  - safe play when winning

In UCT player

- Related: Cazenave + Helmstetter, Combining Tactical Search and Monte-Carlo in the Game of Go, IEEE CIG 2005
- Use static analysis to find achieved goals at root, generate strategies
- Bias playouts to achieve strategies (answer all threats)
- Should help (assuming goals are good)...

UCT(2)

- Can use goals as “virtual moves” in UCT tree
- How does following goal do vs. not following goal in the playouts?