Decomposition Search

A Combinatorial Games Approach to Game Tree Search, with Applications to Solving Go Endgames

Martin Müller
University of Alberta
Edmonton, Canada

Decomposition Search

- What is decomposition search?
- Game decomposition
- Local combinatorial game search (LCGS)
- Application: solving Go endgame studies

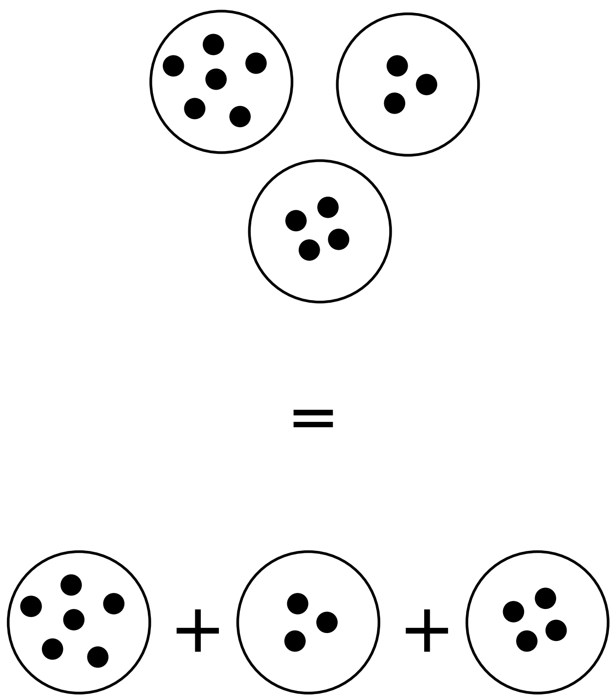
Game Theory	**Solved games**	**Heuristic Programs**
Classical minimax game theory v. Neumann	αβ, PN-search: Gomoku, Nine Men’s Morris,…	Chess, Checkers, Othello,…
Combinatorial game theory Conway, Berlekamp	DS: Go endgame studies	Go

today 2003?

Goal of Decomposition Search

- Compute efficient minimax solutions of decomposable games
- Develop game tree search method that can exploit the power of combinatorial games theory
- Knowledge transfer from mathematical theory to applied AI

Game Decomposition



- Partition game state
- Independence of subgames:
  moves have no affect on other subgames
- Example: Nim - each heap is one subgame

Decomposition Search

- Game decomposition: given game G, find equivalent sum of subgames $G_1 + ... + G_n$
- Local combinatorial game search (LCGS):

  for each G_i, use search to find game graph
- Evaluation: for each game graph, find combinatorial game evaluation $C(G_i)$
- Sum game play: given $C(G_i)$, select an optimal move in $G = G_1 + ... + G_n$

Combinatorial Game Theory

- Developed by Conway, Berlekamp,…
- Abstract definition of two-player games
- Game position defined by sets of follow-up positions for both players
- Books: *On Numbers and Games*, *Winning Ways*
- Main idea: represent game as *sum* of independent subgames

Comparing Game Theories

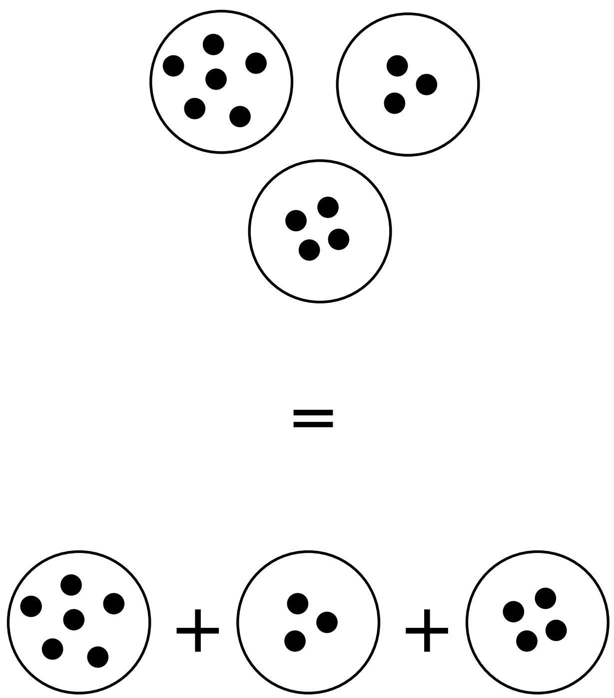
Classical

- Single, monolithic game state
- Full board evaluation
- Single game tree, minimax backup
- Central question: what is the minimax score?

Combinatorial

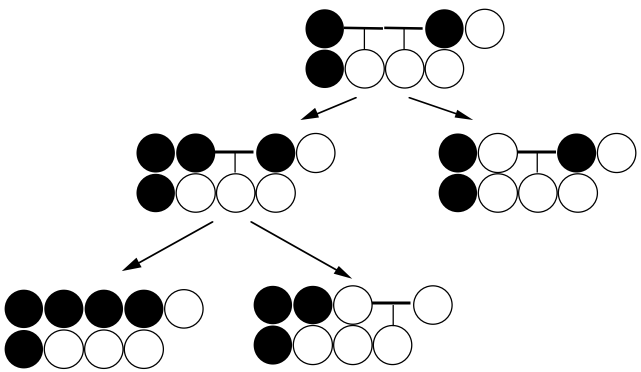
- Partition game into sum of subgames
- Local analysis
- Combination of local results: Algebra of combinatorial games
- Central question: which sums of games are wins?

Example: Nim



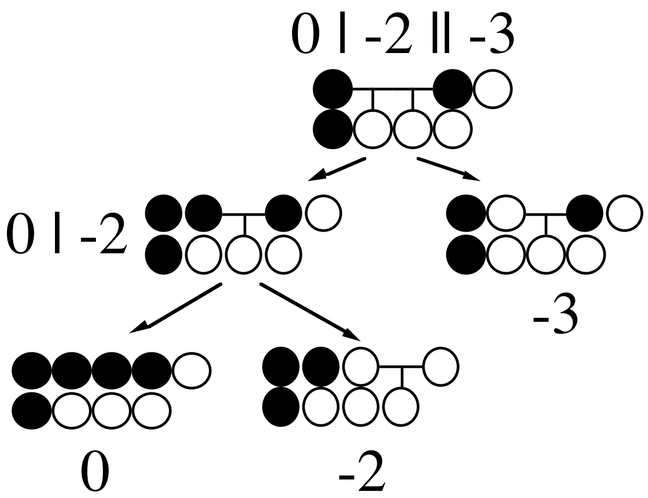
- Full game: high branching factor, long game
- Local game: low branching factor, short game
- Challenge: combine local analyses

Local Combinatorial Game Search



- Search each subgame
- Main difference to minimax search: successive moves by same player
- Local *minimax* search **not enough** to find best global sequence

Local Evaluation



- Evaluate score in leaf nodes
- Backup combinatorial game values
- Value of root completely determines local move values

Sum Game Play

$+_1$

$*$

$-_4|0$

$0|+_2$

$0|-2||-3$ ←

- Given: set of combinatorial game values
- Goal: find best overall move
- Main (fast) method: find dominating move incentive
- Backup (slow) method: use summation of games

Reusing Partial Results

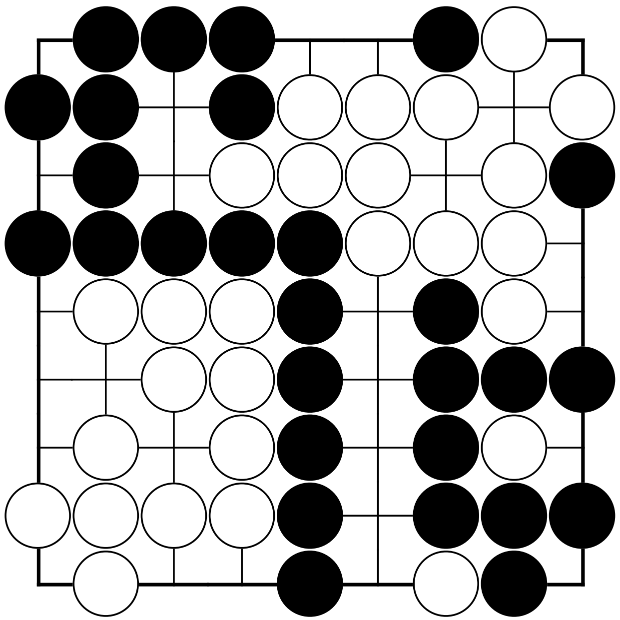
- Advantage of decomposition search: generates useful partial results
- Store evaluated subgames in persistent database
- Database hits speed up search

Complexity of Decomposition Search

Depends on many factors:

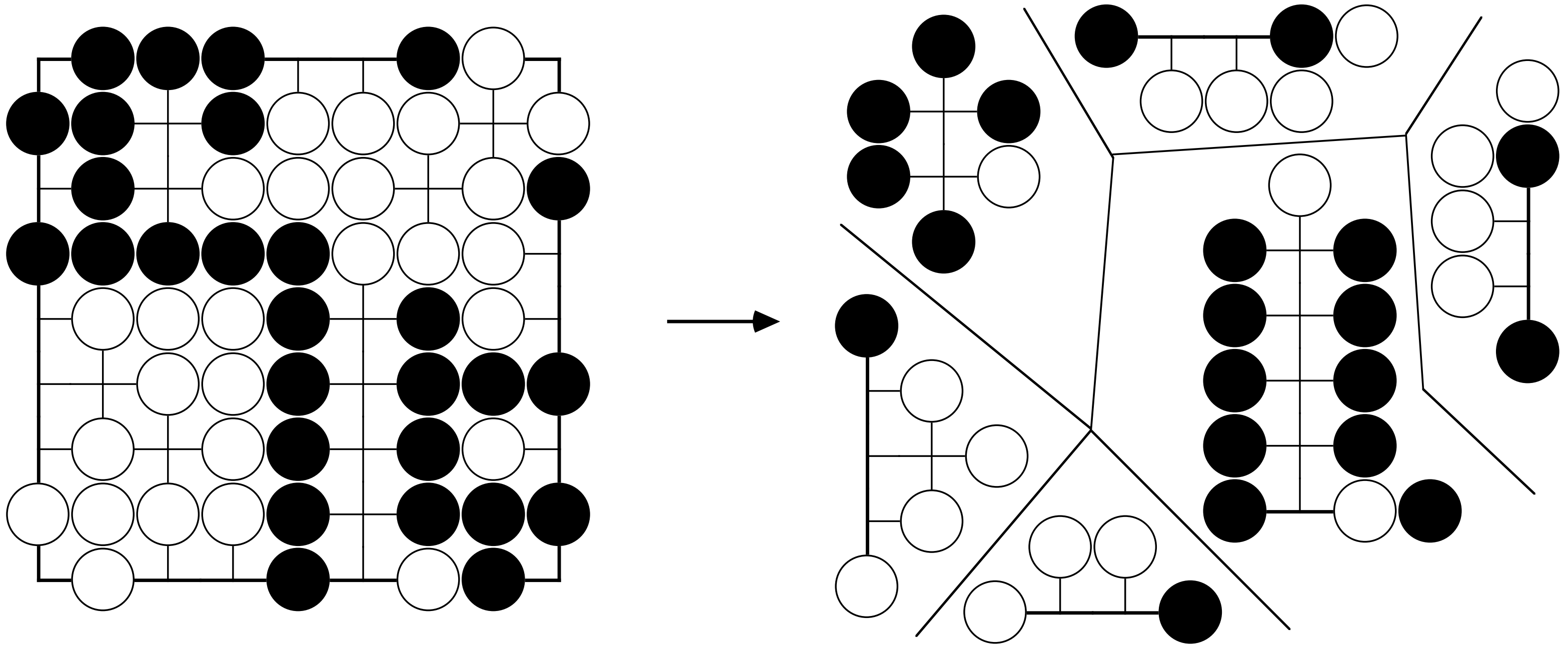
- Game-specific decomposition process
- Size and complexity of subgames
- Type and complexity of combinatorial game expressions
- Existence of a move with dominating incentive
- Complexity of adding subgames

Application to Go Endgames

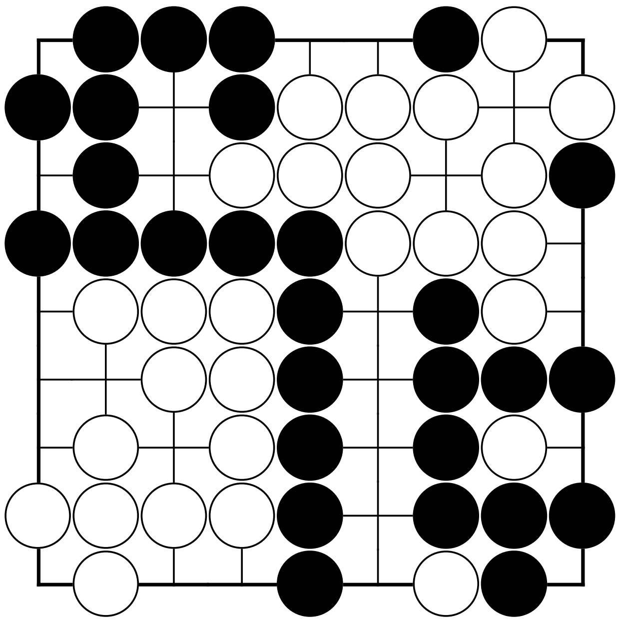


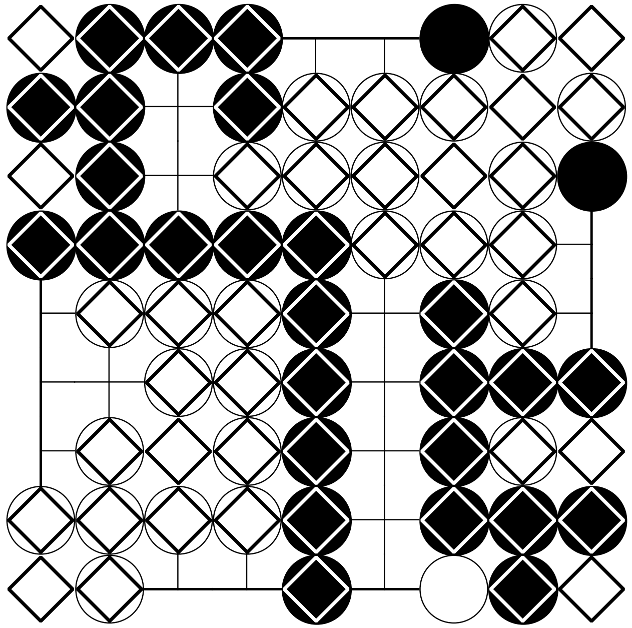
- Decomposition: recognize safe stones and territories
- Local Combinatorial Game Search (LCGS)
- Full board move selection

Board Partition

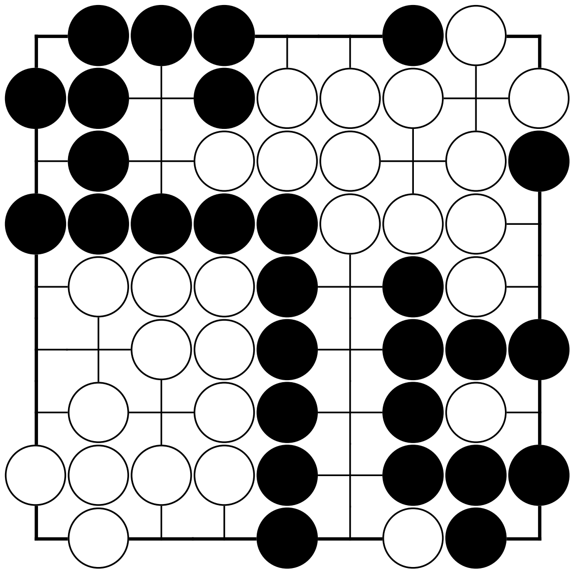


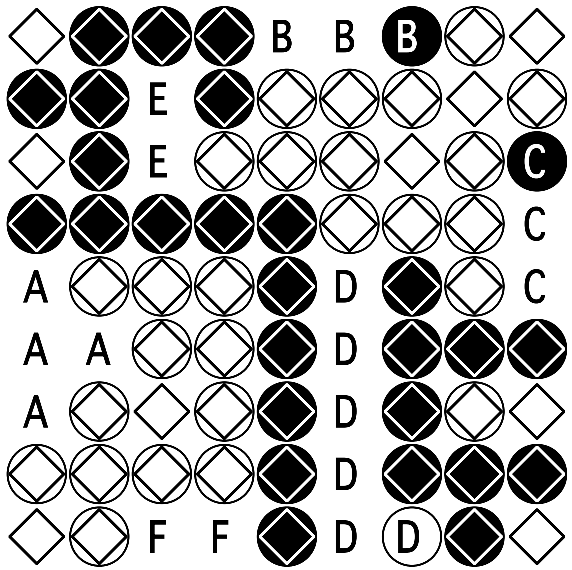
Finding Safe Stones and Territories



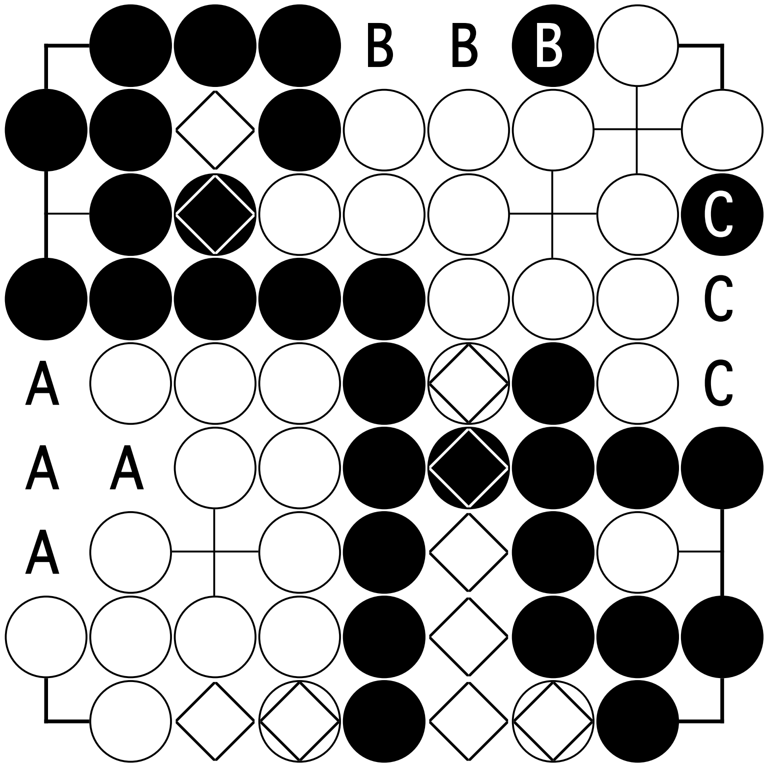


Result of Board Partition



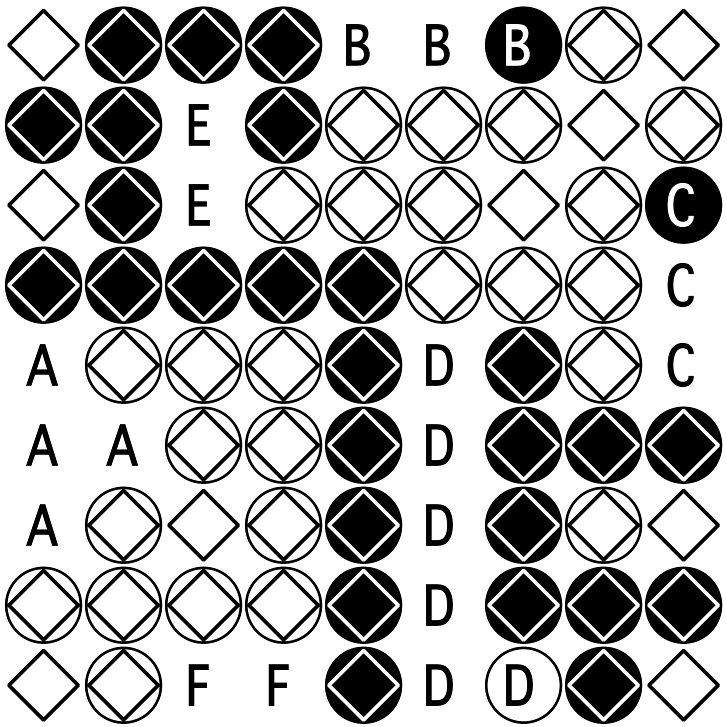


Simplified Endgame Problems



- Simplified Problems: some endgames A..F replaced by constant value territories
- Example: only A, B and C
- D, E, F replaced, played out

Comparison of Decomposition Search and Alpha-beta



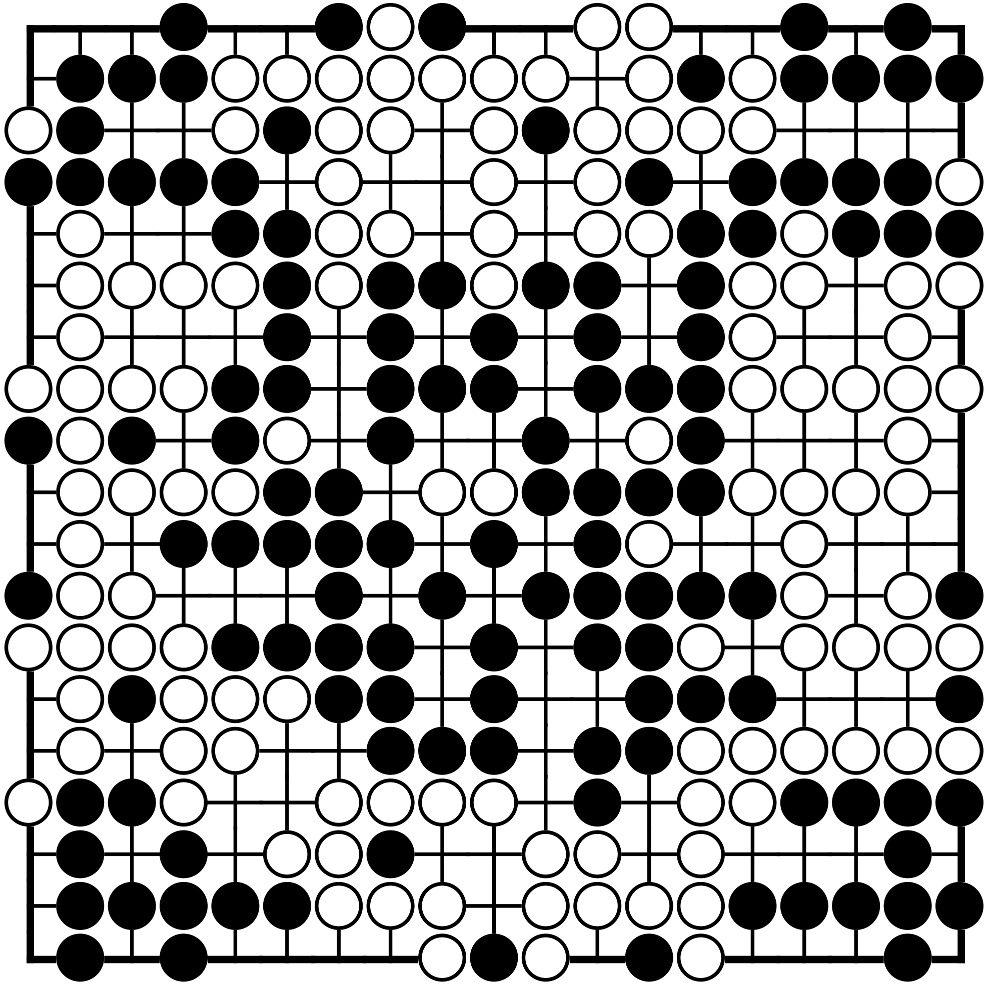
Endgame Area	Size	Nodes	
		DS	αβ
A	4	21	39
A+B	7	26	526
A+B+C	10	31	5905
A+B+C+D	16	42	1097589
A+B+C+D+E	18	45	10243613
A+B+C+D+E+F	20	48	463941123

Evaluation of the Experiment

- Compared decomposition search with full-board alpha-beta
- Alpha-beta: Search time exponential in **size of the full problem**
- Decomposition search: Search time exponential in **size of subproblems**

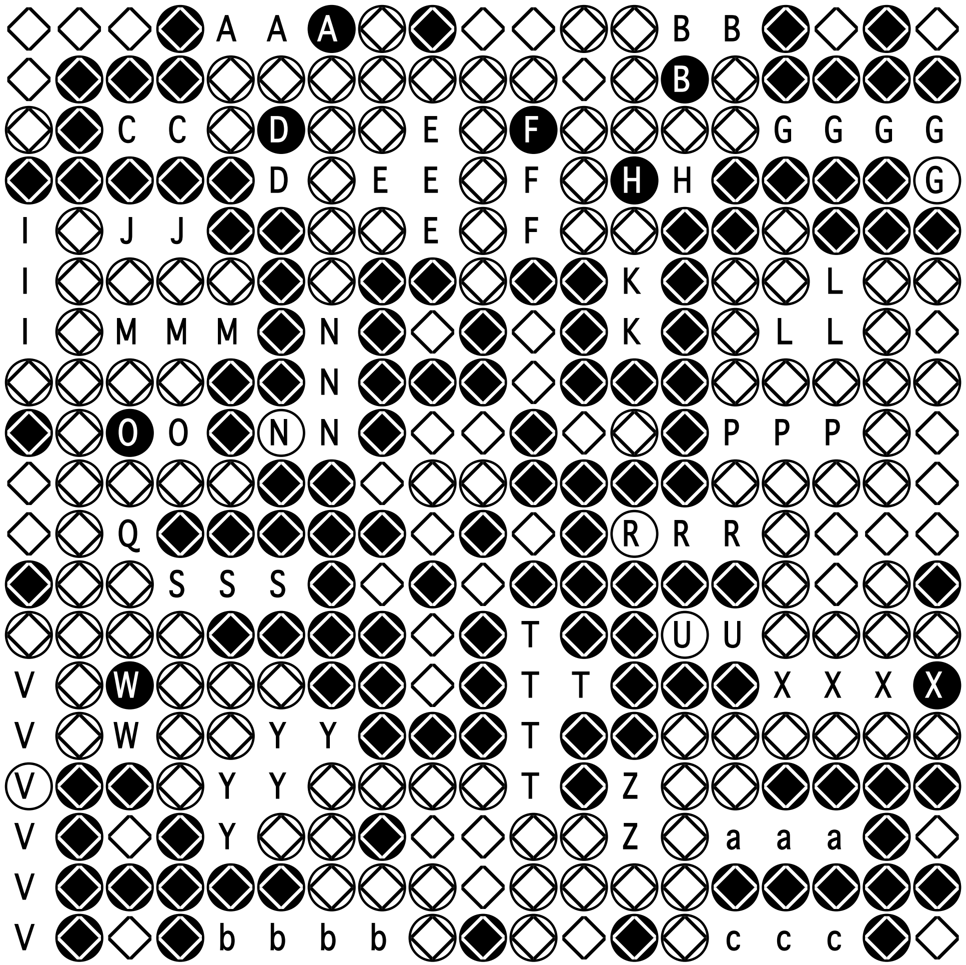
→ Big win for decomposition search!

A 89 Point Problem



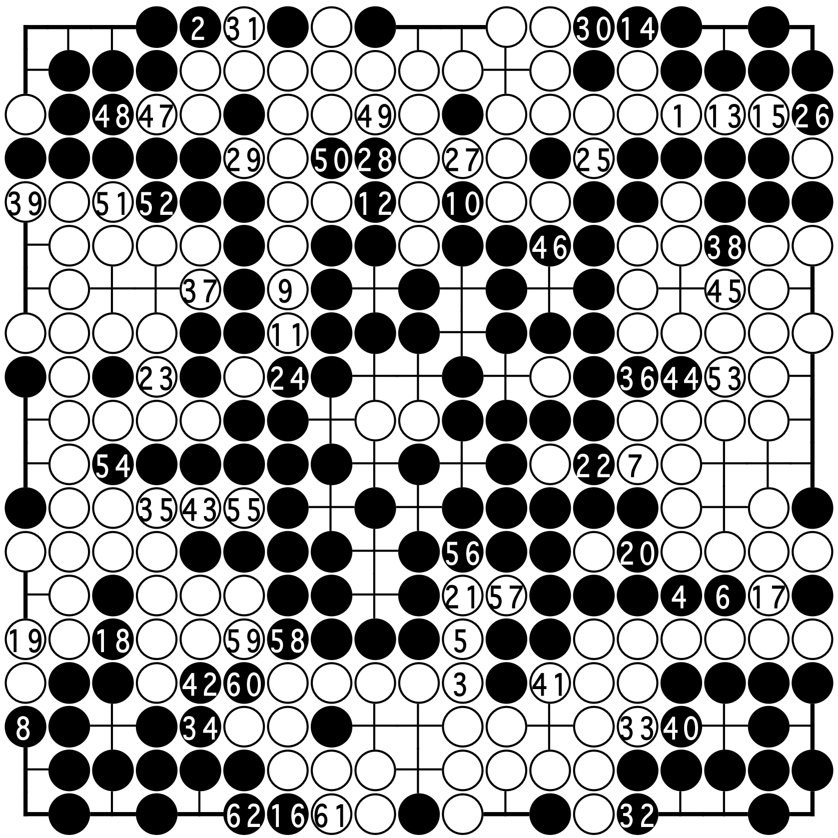
- Problem C.11 of Berlekamp/Wolfe
- 19x19 board
- 29 subgames

89 Point Problem: Partition



- Problem C.11 of Berlekamp/Wolfe
- 19x19 board
- 29 subgames

89 Point Problem: Solution



- Optimal solution computed by DS
- Solution length 62 moves
- 1.1 seconds total time, 0.4 seconds for LCGS
- Generated 420 nodes total for LCGS

Endgame Studies: are they Really Go?

Yes:

- Realistic local game situations
- Real endgame values
- Captures the essence of Go endgame calculation

No:

- All endgames fully independent
- All territories completely safe
- (almost) no Ko
- Real Go endgame mixes endgame calculation and midgame complications

Approximation Algorithms

- Approximation of combinatorial game value: compute thermographs, temperature
- Berlekamp (1996), Spight (1998): theoretical algorithms for thermographs and temperature of games with local position repetition (Ko)
- Müller, Berlekamp and Spight (1996): efficient algorithm for simple repetitions
- No efficient algorithm for general case

Contributions

Decomposition search contributes
to a number of AI research topics:

- Localized processing (multi-agent systems)
- Search methods that propagate more information than just numbers
- Evaluation by partially ordered values

Summary of Decomposition Search

- Goal: solve combinatorial games
- Uses local combinatorial game search
- Globally optimal play
- Works much better than alpha-beta
- Application: solve Go endgame studies
- Future: full scale Go