MCTS Experiments on the Voronoi Game

B. Bouzy, M. Métivier, D. Pellier
bruno.bouzy@paridescartes.fr
http://www.math-info.univ-paris5.fr/~bouzy/

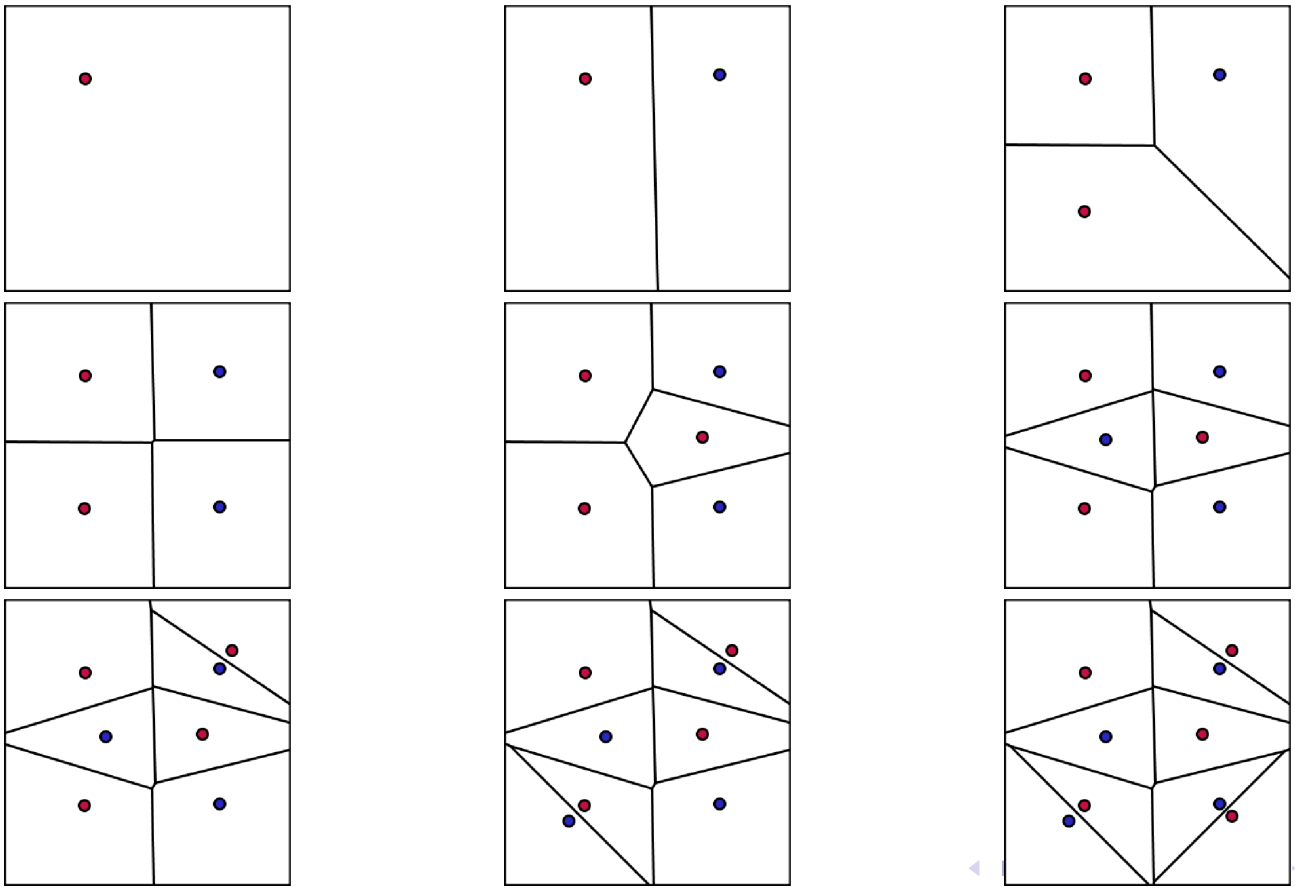
Paris Descartes University

ACG-13, Tilburg, 21st November 2011

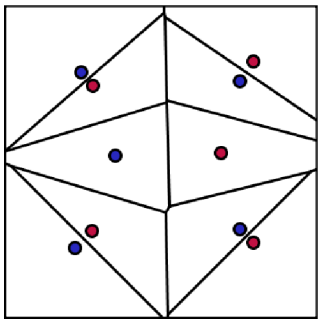
Plan

What is the Voronoi Game ? (1)

- Two players: Red and Blue.
- Red (resp. blue) move = add a red (resp. blue) site, on a point of the square.



What is the Voronoi Game ? (2)



- After each move the Voronoi diagram is computed.
- End of game after a finite number of moves (10).

- Territory game:
  - $redScore = \sum_{redCell} area(redCell)$
  - $blueScore = \sum_{blueCell} area(blueCell)$
  - $score = redScore - blueScore$
  - Red wins if $score > komi$
- $komi < 0$
  - Blue has the advantage of playing the last move

Why study the Voronoi Game ?

- Actions in $]0,1[\times]0,1[$: infinite branching factor
- Territory game
- Relatively new game
  - Applets (Faidley 2004, Selimi 2008)
  - Papers (Ahn 2004, Anuth 2007)
  - One-round (Cheong 2002, Fekete 2005)
- MCTS, UCT mostly applied to finite branching factor games
- Infinite arm number methods exist:
  - Gaussian Processes (GP) (Rasmussen & Williams 2006),
  - Hierarchical Optimistic Optimization (HOO) (Bubeck & al. 2011).

MCTS, UCT, RAVE

- With a finite branching factor:
- D discretisation of $]0,1[\times]0,1[$, finite but large, defined a priori.
- UCT (Kocsis Szepesvari 2006) has 4 stages:
  - Tree selection
    - $x_{next} = \arg\max_{x \in \{x_1,\ldots,x_k\}} (\mu(x) + ucb(x))$
    - $ucb(x_i) = \sqrt{C \times \frac{log(T)}{N_i}}$
  - Expansion
  - Simulation
  - Back-propagation
- RAVE (Gelly 2007)
  - $m_{RAVE} = \beta \times m + (1 - \beta) \times m_{AMAF}$
  - $\beta = \sqrt{\frac{N_i}{K+N_i}}$
- With an infinite branching factor, how to perform UCT selection ?

Gaussian Processes Principle

- Goal: optimize an unobserved noisy target function f.
- At time step, choose x and observe $f(x)$ with $x \in X$ infinite.
- At time t, x_i with $1 \leq i \leq N$ have been observed.
- At time $t+1$, which point to observe ?
- Theoretical solution: UCB-GP or GP (Rasmussen 2006).
- Cost: matrix inversion
- Unbearable.

Our Light Gaussian Processes (LGP)

- $x \in D$ with D finite but large. D defined offline.
- Acquisition function $A(x) = \widehat{f}(x) + g(x)$.
- On observed points x_i:
  - $\widehat{f}(x_i) = \mu(x_i)$.
  - $g(x_i) = ucb(x_i)$.
- On unobserved points x:
  - $\widehat{f}(x) = \frac{\sum_{i=1}^{N} \mu(x_i) \times \exp(-a(x-x_i)^2)}{\sum_{i=1}^{N} \exp(-a(x-x_i)^2)}$
  - $g(x) = G \times \prod_{i=1}^{N} (1 - \exp(-b(x-x_i)^2))$
  - with a, b, G parameters.
- At time $t+1$, $x = \arg\max_{x \in D} A(x)$
- Performed at each node selection of UCT.
- Cost: browse D
- Bearable.

Calibration experiments

- Goal: tune the parameters of each algorithm.
- When? Before comparing the algorithms.
- List of parameters to be tuned: C, K, G, a, b, D, $komi$, Ns.
- How? In self-play.
- Effort: 90% of the total effort.
- Finally: $C = 0.25$, $K = 400$, $G = 2$, $a = b = 180$, $15 \times 15 \leq |D| \leq 25 \times 25$, $komi \in [-0.04, -0.08]$, $Ns = 4000$.

Results of experiments without knowledge

- *Gamelength* = 10, UCT, RAVE, LGP, RLGP (RAVE+LGP).
- 100 game result between Red (line) and Blue (column)
- T: total number of wins, *komi* = -0.08, *Ns* = 4000.

	UCT	*RAVE*	*LGP*	*RLGP*	T
UCT	39-61	30-70	43-57	46-54	350
RAVE	51-49	56-44	53-47	51-49	399
LGP	63-37	63-37	78-22	75-25	428
RLGP	55-45	63-37	77-23	80-20	423

- LGP is the best.
- 2 statistically significant comparisons:
  - RAVE vs UCT : 60%-40%. $\sigma = 3\%$
  - LGP vs UCT : 60%-40%. $\sigma = 3\%$
- RAVE and LGP do not add up each other.

Considering domain-dependent knowledge

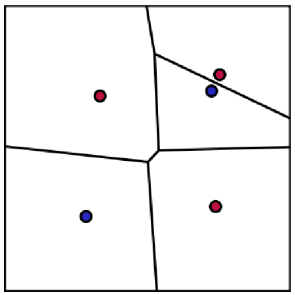
- Jens Anuth master thesis (2007)
- Last move.
- The Biggest Cell Attack (BCA) player.
- Balance. The Defensive and Balanced (DB) player.
- Balanced UCT (B-UCT).
- B-UCT plus (simple) attacks = aB-UCT.
- One-round VG.
  - The 2nd Player strategy in the one-round VG (1R2P).
- B-UCT with (sophisticated) attacks = AB-UCT.

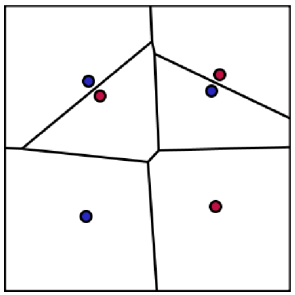
The (before) last move special case(s)

- Last move: depth-one minimax search
- Appropriate discretisation
- Before last move: depth-two minimax search

The BCA Player

- Key concepts: balance and gravity center G of a cell.
- Balanced cell: site $S = G$, unbalanced otherwise.
- Biggest Cell Attack (BCA) player:
  - plays on the gravity center of the biggest cell of the opponent.

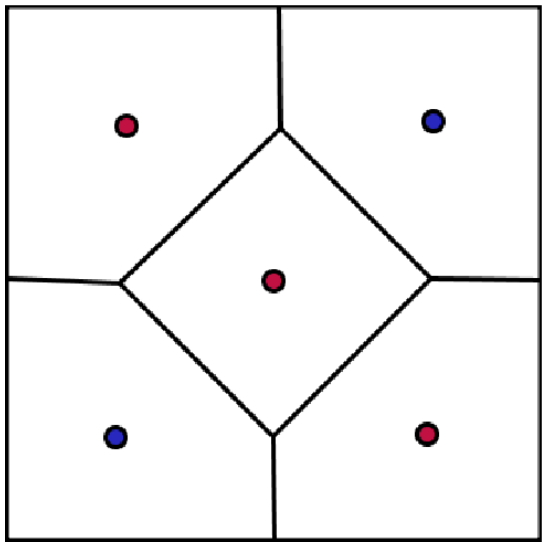




- Effective against players using unbalanced cells.
- Weakness: produces unbalanced cells.
- BCA vs Depth-one(D=20x20) : 60%-40%. $\sigma = 5\%$

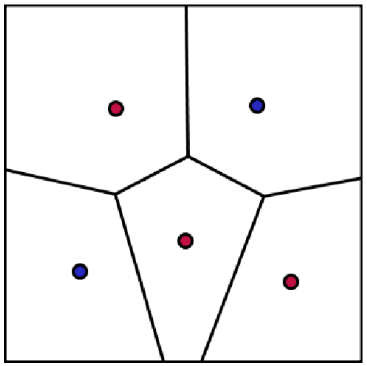
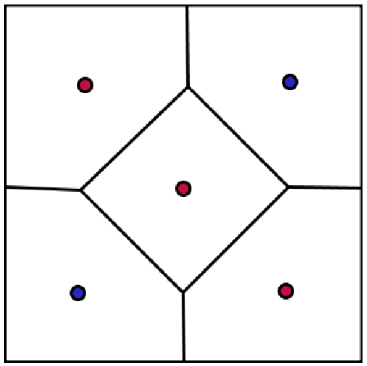
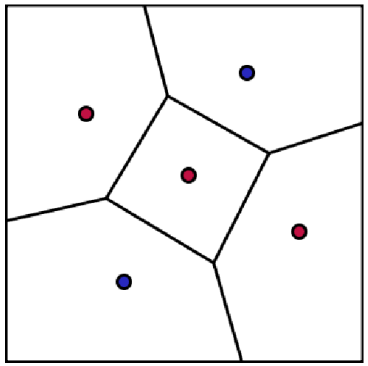
Balance

- Importance of "key points" (Ahn & al 2004).
- In a $2 \times N$ game, they are N key points.
- Playing N sites on the N key points yields N balanced cells.



The Lloyd algorithm

- The Lloyd algorithm (Lloyd 1982)
  - Input: S set of sites
  - repeat
    - V = Voronoi Diagram(S)
    - For each cell, move its site s to its gravity center g
  - until V balanced enough
- 3 outputs:

The Defensive and Balanced player (DB)

- A n-site balanced diagram is computed offline.
- The DB player follows the n-site balanced diagram.
- (DB uses the last move special case.)
- (Except for the last move,) DB plays instantly.
- DB vs UCT(Ns=1000) : 95%-5%. $\sigma = 2\%$
- DB vs UCT(Ns=4000) : 70%-30%. $\sigma = 3\%$

The Balanced UCT Player (B-UCT)

- Balanced UCT (B-UCT): UCT with almost balanced simulations.
  - Compute *RedDB* (resp. *BlueDB*) the most balanced defensive diagram of Red (resp. Blue).
  - In the simulations:
    - For Red, follow *RedDB* with small noise.
    - For Blue, follow *BlueDB* with small noise.
- Adapt the Lloyd algorithm:
  - output diagrams are not balanced anymore.
  - unbalance of a diagram $= \max_{cell} d(site(cell), G(cell))$.
- B-UCT(Ns=4000) vs UCT(Ns=4000) : 85%-15%. $\sigma = 3\%$
- B-UCT(Ns=4000) vs DB : 25%-75%. $\sigma = 3\%$

The aB-UCT Player

- attack B-UCT (aB-UCT): B-UCT using BCA in the simulations.
  - With probability A^{1-L}, the BCA strategy is used.
  - L: number of remaining moves, $A = 2$ (after calibration).
- Against DB:
  - aB-UCT(Ns=500) vs DB : 55%-45%. $\sigma = 5\%$
- The BCA attack is not effective against balanced diagrams.
- Next: how to attack a balanced diagram of the opponent ?

The 1 Round 2nd Player (1R2P) Strategy

- Consider the one-round VG (Fekete & Meijer 2005):
  - Red plays all his sites "in one round", then Blue plays all his sites.
  - Assume V is the red diagram after red move in the one-round game,
  - Perform a depth-one search to find the first blue site,
  - Use BCA to find the $N-1$ other blue sites.
- The 1-round 2nd player (1R2P) strategy:
  - $B = 0$
  - repeat
    - play the depth-one strategy B times
    - play the BCA strategy $N - B$ times
    - $B = B + 1$
  - until success or $B > N$
- *Aggressive*(V): the N-site diagram that attacks the N-site V.

The AB-UCT Player

- Attack B-UCT (AB-UCT): B-UCT + aggressive diagrams.
  - Compute *RedDB* and *BlueDB*.
  - Compute *Aggressive*(*RedDB*) and *Aggressive*(*BlueDB*).
  - In the simulations:
    - For Red, follow *RedDB* or *Aggressive*(*BlueDB*)) with noise.
    - For Blue, follow *BlueDB* or *Aggressive*(*RedDB*)) with noise.
- The results:
  - AB-UCT(Ns=500) vs aB-UCT(Ns=500) : 71%-29%. $\sigma = 3\%$
  - AB-UCT(Ns=500) vs DB : 77%-23%. $\sigma = 3\%$
  - AB-UCT(Ns=500) vs UCT(Ns=500) : 93%-7%. $\sigma = 2\%$

Against human players and against other work

- Human players (H)
  - Good at roughly seeing the balance of a diagram,
  - With current GUI, bad at clicking precisely somewhere.
  - Limited to a rank inferior to *DB*.
- Jens Anuth master thesis (2007)
  - depth-one search over clever evaluation functions.
  - arbitrary shapes of polygons.
  - key point strategy (*DB*).
- Summary: UCT $\leq$ H $\leq$ BCA $\ll$ DB $\ll$ AB-UCT.

Conclusion

- First work using MCTS (UCT) on the Voronoi Game.
- MCTS + Voronoi Game + enhancements: RAVE, GP.
- MCTS + Voronoi game + domain dependent knowledge (balance)
  - BCA
  - DB
  - B-UCT
  - aB-UCT
  - AB-UCT
- Future work
  - enhance AB-UCT.
  - try HOO (Bubeck 2011) instead of GP.
  - multi-player VG.
  - popularize VG.

Thank you for your attention!