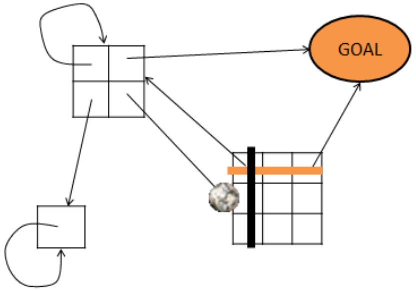


Real Algebraic Geometry in Computational Game Theory

Peter Bro Miltersen

Aarhus University

CTIC
交互计算
ctic.au.dk

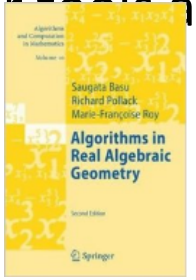
Solving Polynomial Equations, Berkeley, 15/10/14

Computational Game Theory

- Input: Description of *game.*
- Output: *Solution* to game.
  - Find value/minimax strategy
  - Find Nash equilibrium
  - …..

R.A.G. in ``pure´´ game theory

- Long history
- Classics in the theory of *stochastic* games:
  - Truman Bewley and Elon Kohlberg. *The asymptotic theory of stochastic games.* Mathematics of Operations Research, 1:197-208, 1976.
  - J.F. Mertens and A. Neyman. *Stochastic games.* Int. J. of Game Theory, pages 53-66, 1981.
  - Emanuel Milman. *The Semi-Algebraic Theory of Stochastic Games.* Mathematics of Operations Research 27:2 , 401-418, 2002.
  - A. Neyman. *Real Algebraic tools in Stochastic Games*. Stochastic Games and Applications. NATO Science Series Volume 570, 2003, pp 57-75
- Often relies on ``crude´´ tools (e.g. Tarski Transfer Principle)
- ***Slogan of this talk:*** In the computational setting, finer tools are advantageous.



Saugata Basu
Richard Pollack
Marie-Françoise Roy
Algorithms in Real Algebraic Geometry

Recent papers

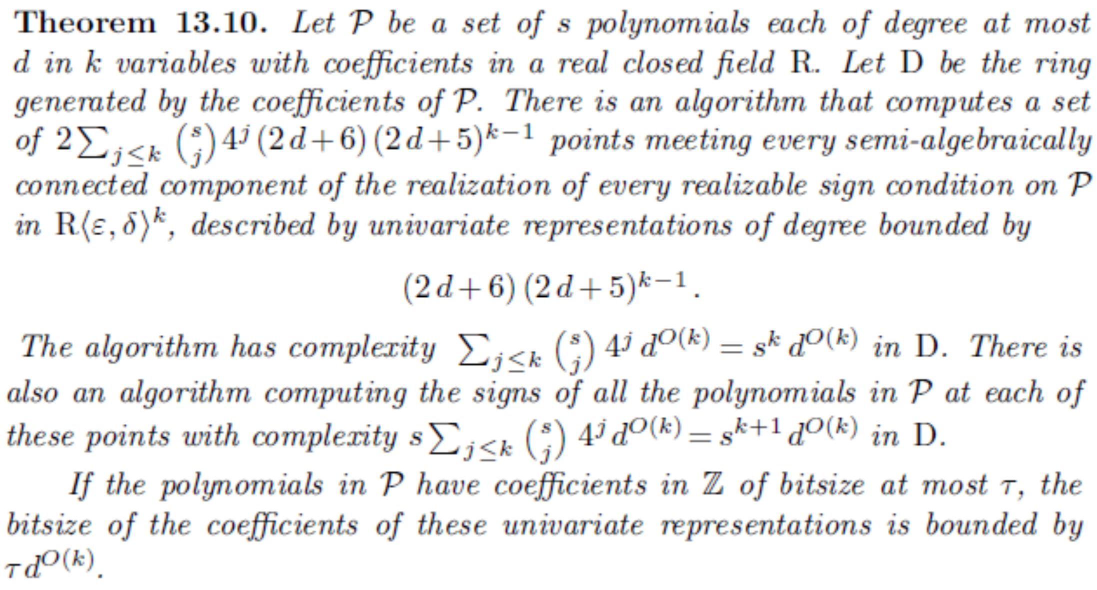
- Kristoffer Arnsfelt Hansen, Michal Koucký, and Peter Bro Miltersen. **Winning concurrent reachability games requires doubly exponential patience**. In *Proceedings of LICS'09*, pages 332–341.
- Kristoffer Arnsfelt Hansen, Rasmus Ibsen-Jensen, and Peter Bro Miltersen**. The complexity of solving reachability games using value and strategy iteration.** In *Proceedings of CSR'11*, volume 6651 of LNCS, pages 77–90.
- Kristoffer Arnsfelt Hansen, Michal Koucký, Niels Lauritzen, Peter Bro Miltersen, and Elias P. Tsigaridas. **Exact algorithms for solving stochastic games .** In *Proceedings of STOC'11,* pages 205–214.
- Søren Kristoffer Stiil Frederiksen and Peter Bro Miltersen. **Approximating the value of a concurrent reachability game in the polynomial time hierarchy**. In *Proceedings of ISAAC'13*, volume 8283 of LNCS, pages 457–467.
- Søren Kristoffer Stiil Frederiksen and Peter Bro Miltersen. **Monomial strategies for concurrent reachability games and other stochastic games**. In *Proceedings of RP'13*, volume 8169 of LNCS, pages 122–134.
- Kousha Etessami, Kristoffer Arnsfelt Hansen, Peter Bro Miltersen, Troels Bjerre Sørensen. **The complexity of approximating a trembling hand perfect equilibrium of a multi-player game in strategic form**. In *Proceedings of SAGT'14*, volume 8768 of Lecture Notes in Computer Science, volume 8768, pages 231-243, 2014.

Today: Just one example

- Computing the value of a *concurrent reachability game.*
  - Worst case time complexity analysis of *Strategy iteration* algorithm.
  - [HKM'09, HKLMT'11, HIM'11]

- Algorithm does ***not*** rely on R.A.G.
- Quantitative but ***not*** algorithmic R.A.G. needed.

R.A.G. engine: The Sampling Theorem

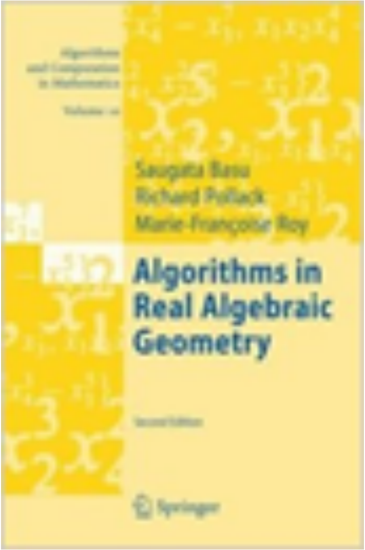


Theorem 13.10. *Let $\mathcal{P}$ be a set of s polynomials each of degree at most d in k variables with coefficients in a real closed field* R. *Let* D *be the ring generated by the coefficients of $\mathcal{P}$. There is an algorithm that computes a set of $2\sum_{j\leq k}\binom{s}{j}4^j(2d+6)(2d+5)^{k-1}$ points meeting every semi-algebraically connected component of the realization of every realizable sign condition on $\mathcal{P}$ in $\mathrm{R}\langle\varepsilon,\delta\rangle^k$, described by univariate representations of degree bounded by*

$$(2d+6)(2d+5)^{k-1}.$$

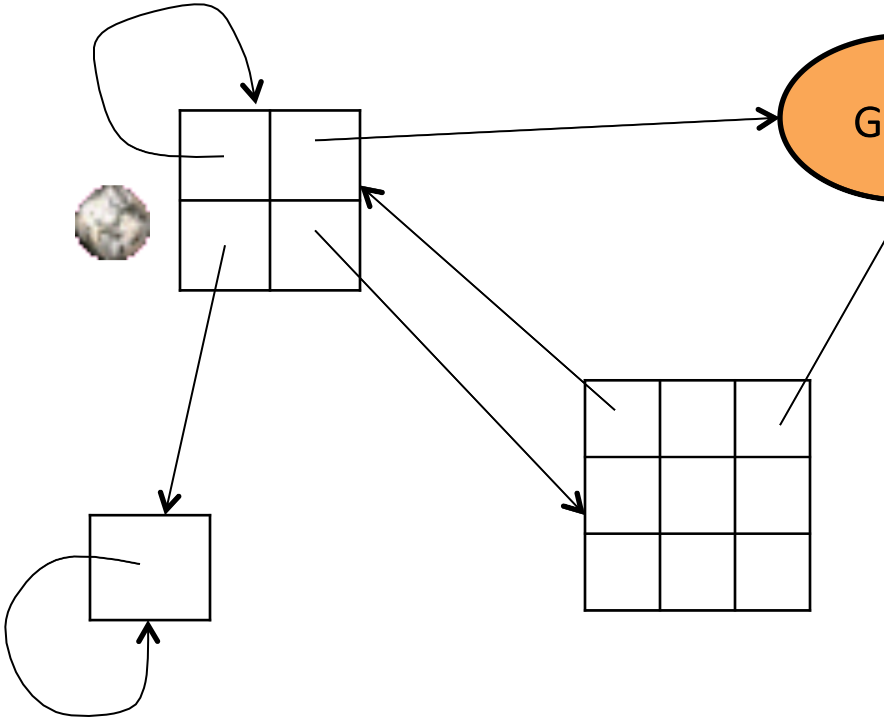
The algorithm has complexity $\sum_{j\leq k}\binom{s}{j}4^j d^{O(k)} = s^k d^{O(k)}$ in D. *There is also an algorithm computing the signs of all the polynomials in $\mathcal{P}$ at each of these points with complexity $s\sum_{j\leq k}\binom{s}{j}4^j d^{O(k)} = s^{k+1}d^{O(k)}$ in* D.

If the polynomials in $\mathcal{P}$ have coefficients in $\mathbb{Z}$ of bitsize at most τ, the bitsize of the coefficients of these univariate representations is bounded by $\tau d^{O(k)}$.



If a sign condition is realizable, then it is realized by a point of ”low algebraic complexity”.

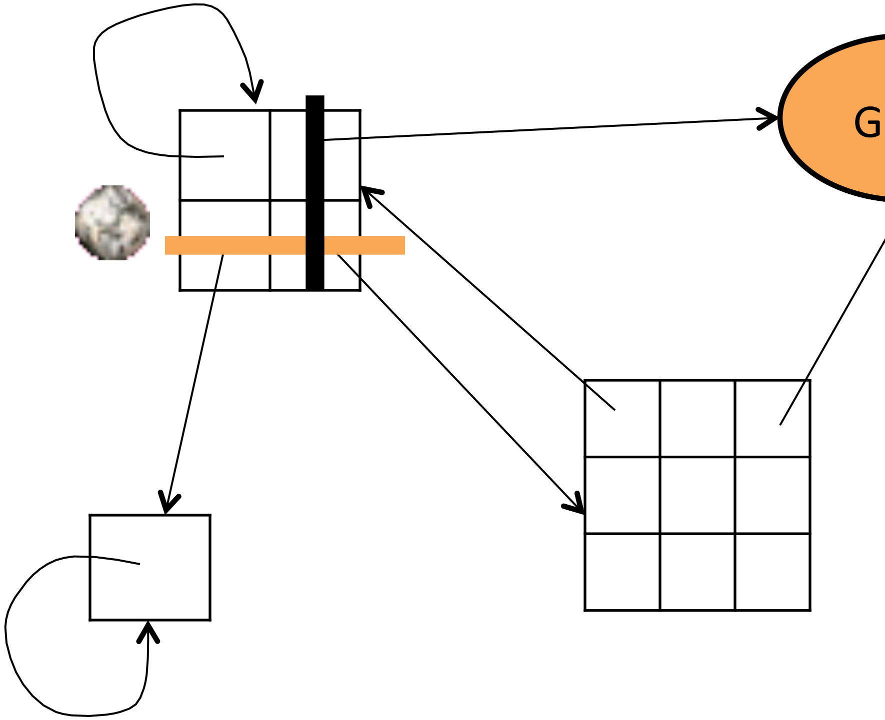
Concurrent Reachability Game (CRG)



Row player wants pebble to reach GOAL

Column player wants to prevent pebble from reaching GOAL

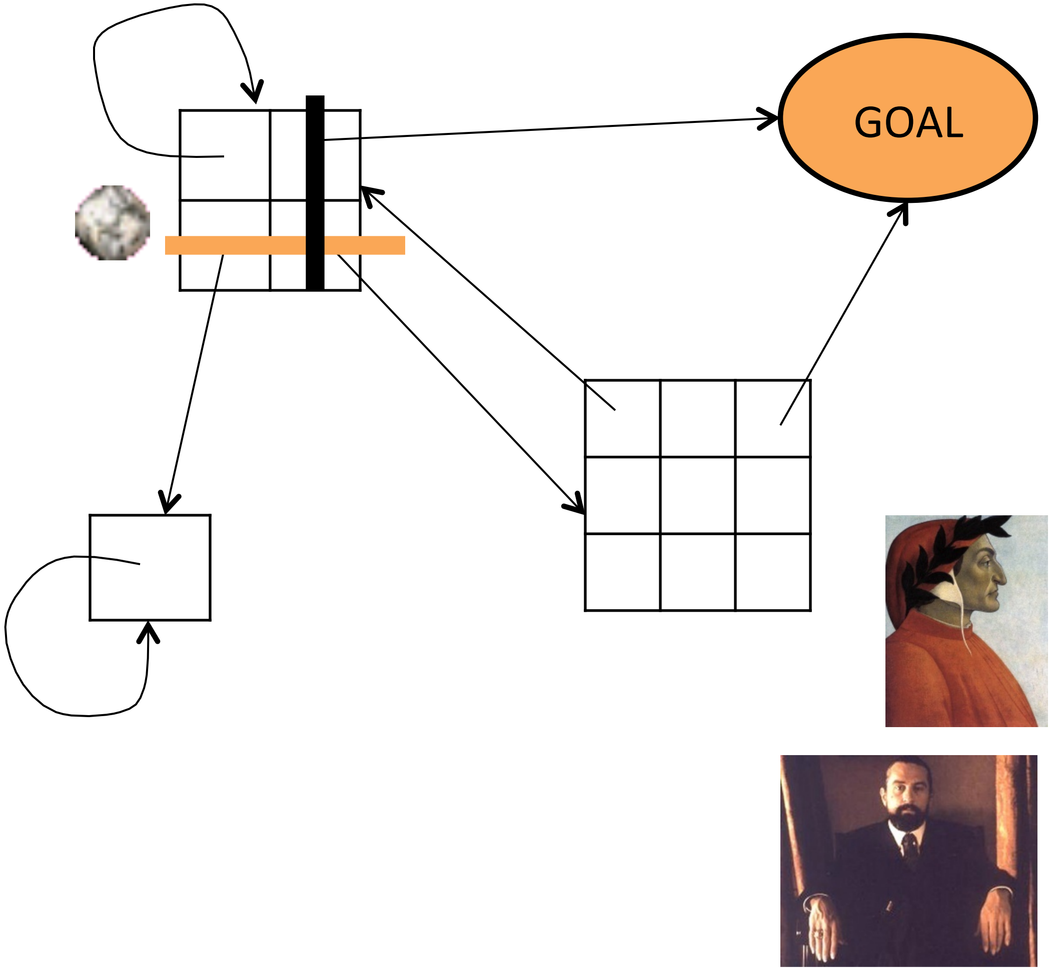
Concurrent Reachability Game (CRG)



Row player wants pebble to reach GOAL

Column player wants to prevent pebble from reaching GOAL

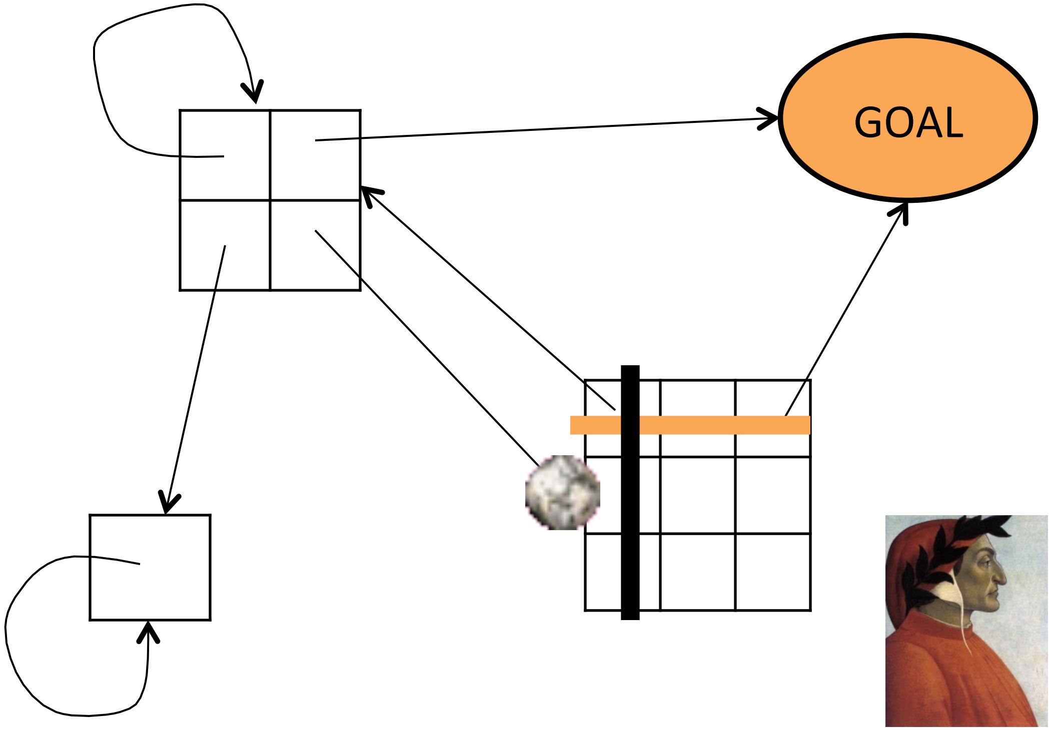
Concurrent Reachability Game (CRG)



Row player wants pebble to reach GOAL

Column player wants to prevent pebble from reaching GOAL

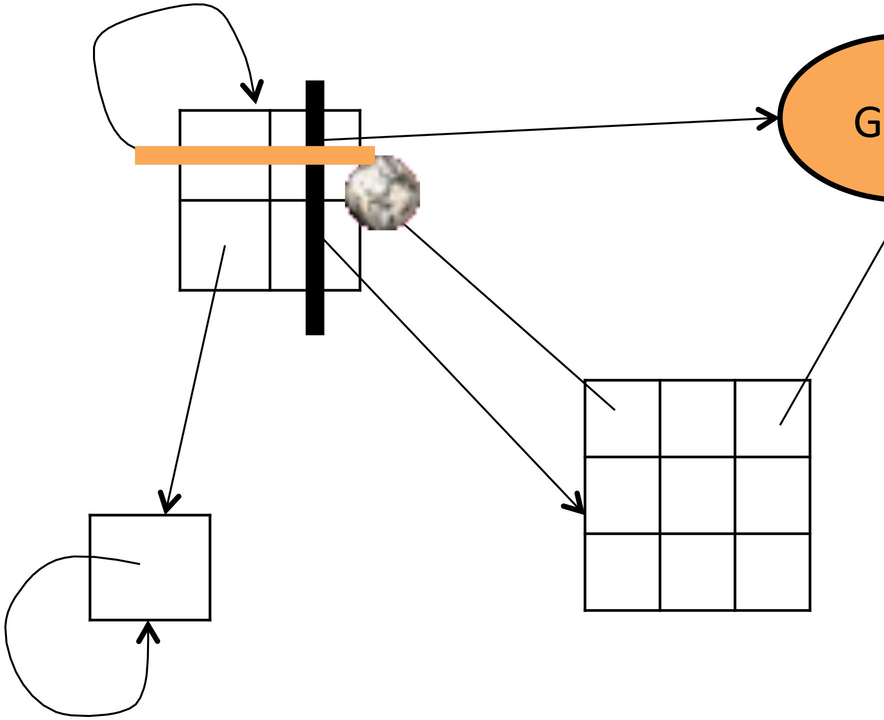
Concurrent Reachability Game (CRG)



Row player wants pebble to reach GOAL

Column player wants to prevent pebble from reaching GOAL

Concurrent Reachability Game (CRG)




Row player wants pebble to reach GOAL


Column player wants to prevent pebble from reaching GOAL

Values and Near-Optimal Strategies (Everett'57)

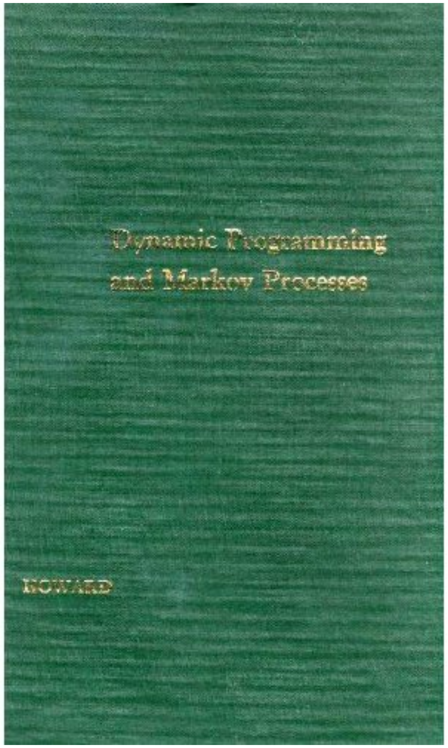
- Each position i in a CRG has a *value* v_i so that

$$\begin{aligned} v_i &= \min_{\text{stationary } \mathbf{y}} \max_{\text{general } \mathbf{x}} \mu_i(\mathbf{x},\mathbf{y}) \\ &= \mathbf{sup}_{\text{ stationary } \mathbf{x}} \min_{\text{general } \mathbf{y}} \mu_i(\mathbf{x},\mathbf{y}) \end{aligned}$$

where $\mu_i(\mathbf{x},\mathbf{y})$ is the probability of reaching GOAL when row player plays by strategy **x** and column player plays by strategy **y**.

Howard's algorithm (1960)

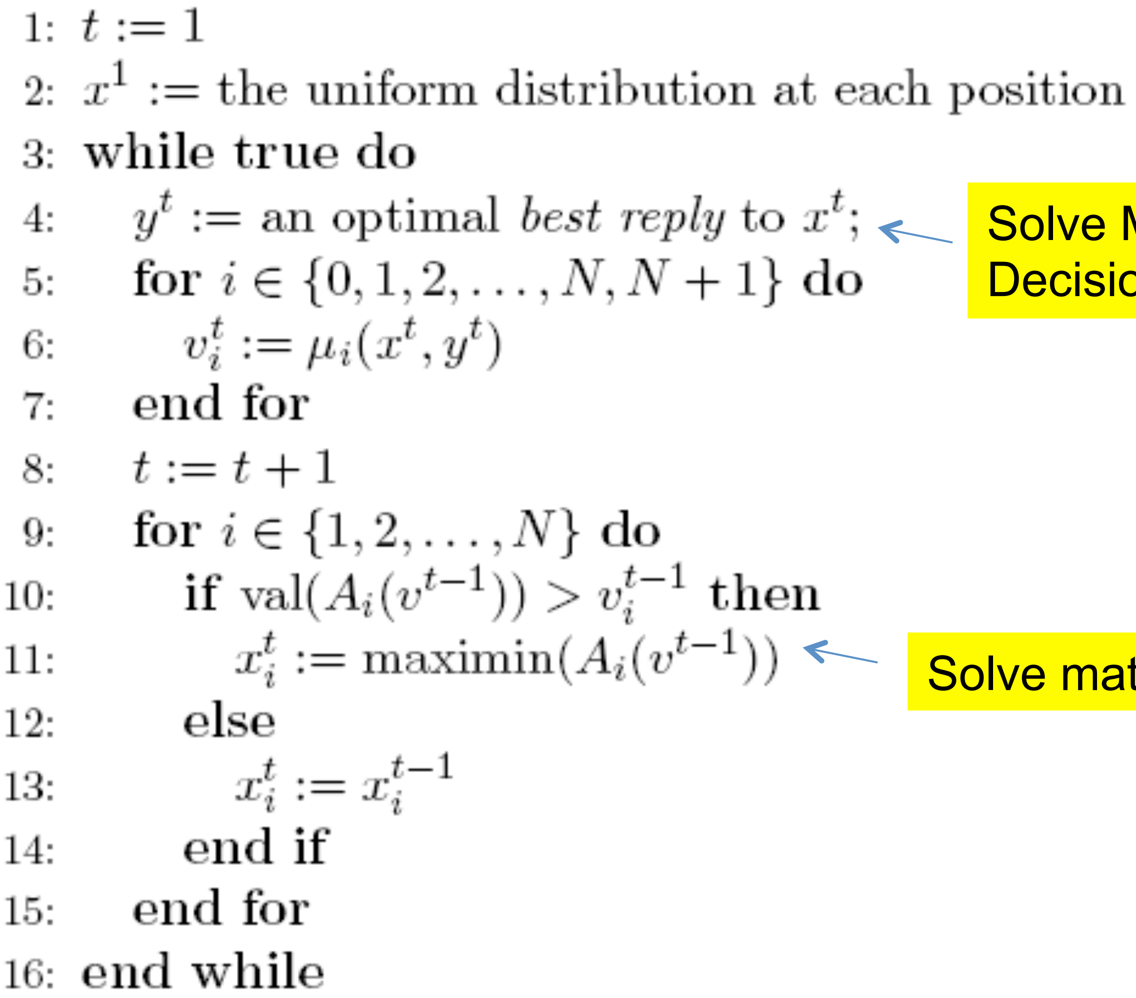
(aka policy iteration, policy improvement, strategy iteration/improvement)



Basic algorithm for online, sequential decision making in face of uncertainty

Howard’s algorithm for CRGs

Chatterjee, de Alfaro, Henzinger ’06, Etessami and Yannakakis ‘06



```
$t := 1$
$x^1$ := the uniform distribution at each position
while true do
    $y^t$ := an optimal best reply to $x^t$;
    for $i \in \{0, 1, 2, \ldots, N, N+1\}$ do
        $v_i^t := \mu_i(x^t, y^t)$
    end for
    $t := t + 1$
    for $i \in \{1, 2, \ldots, N\}$ do
        if $\mathrm{val}(A_i(v^{t-1})) > v_i^{t-1}$ then
            $x_i^t := \mathrm{maximin}(A_i(v^{t-1}))$
        else
            $x_i^t := x_i^{t-1}$
        end if
    end for
end while
```

Solve Markov Decision Process

Solve matrix game

Properties

- The valuations v^t_i converge to the values v_i (from below).
- The strategies x^t guarantee the valuations v^t_i for row player.

- ***What is the number of iterations required to guarantee a good approximation?***

Main theorem

For all games with N positions and m actions for each player in each position, $(1/\varepsilon)^{m^{O(N)}}$ iterations is sufficient to arrive at ε-optimal strategy.

N = Number of positions

m = dimension of (largest) matrix

Step 1: Reduction to analysis of value iteration

- We can relate the valuations computed by strategy iteration to the valuations computed by ***value iteration***.

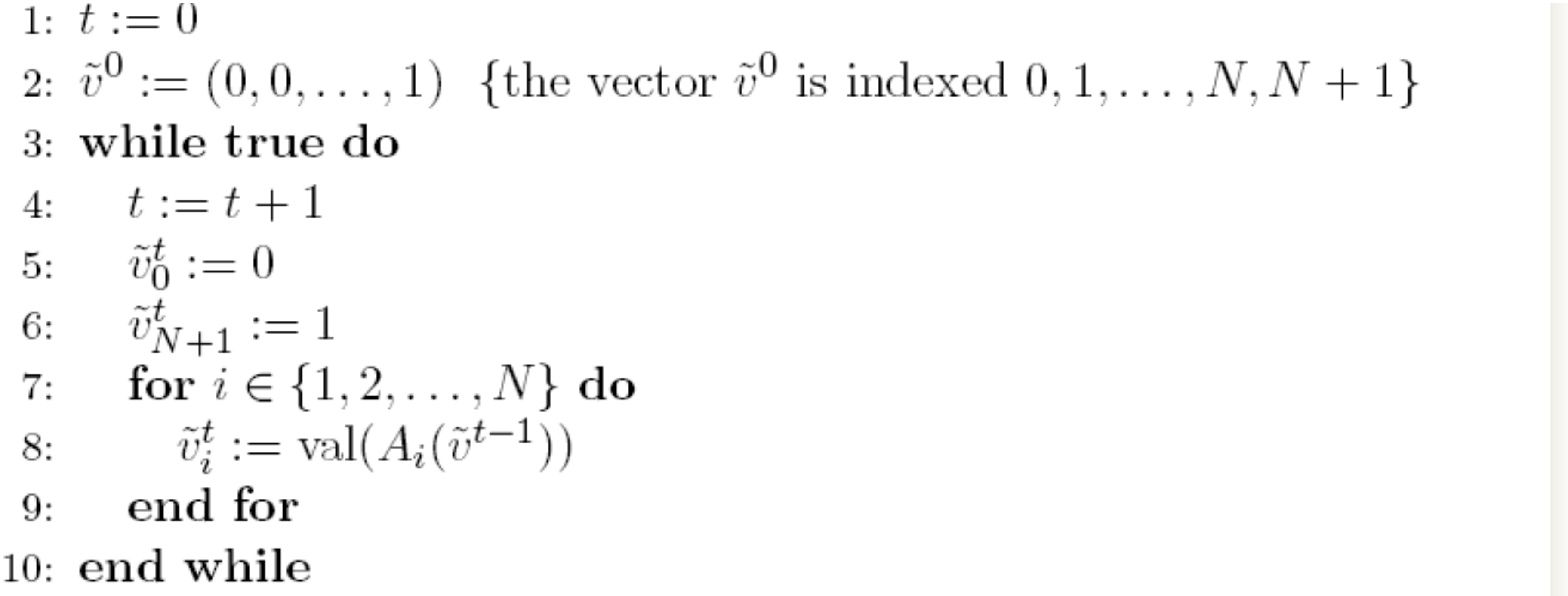
$$\tilde{v}_i^t \leq v_i^t \leq v_i$$

Actual values

Valuations computed by value iteration

Valuations computed by strategy iteration

Value iteration (dynamic programming)



```
$t := 0$
$\tilde{v}^0 := (0, 0, \dots, 1)$  {the vector $\tilde{v}^0$ is indexed $0, 1, \dots, N, N+1$}
while true do
    $t := t + 1$
    $\tilde{v}^t_0 := 0$
    $\tilde{v}^t_{N+1} := 1$
    for $i \in \{1, 2, \dots, N\}$ do
        $\tilde{v}^t_i := \mathrm{val}(A_i(\tilde{v}^{t-1}))$
    end for
end while
```

Value iteration computes the value of a time bounded game,
for larger and larger values of the time bound *t*, by ***backward induction***.

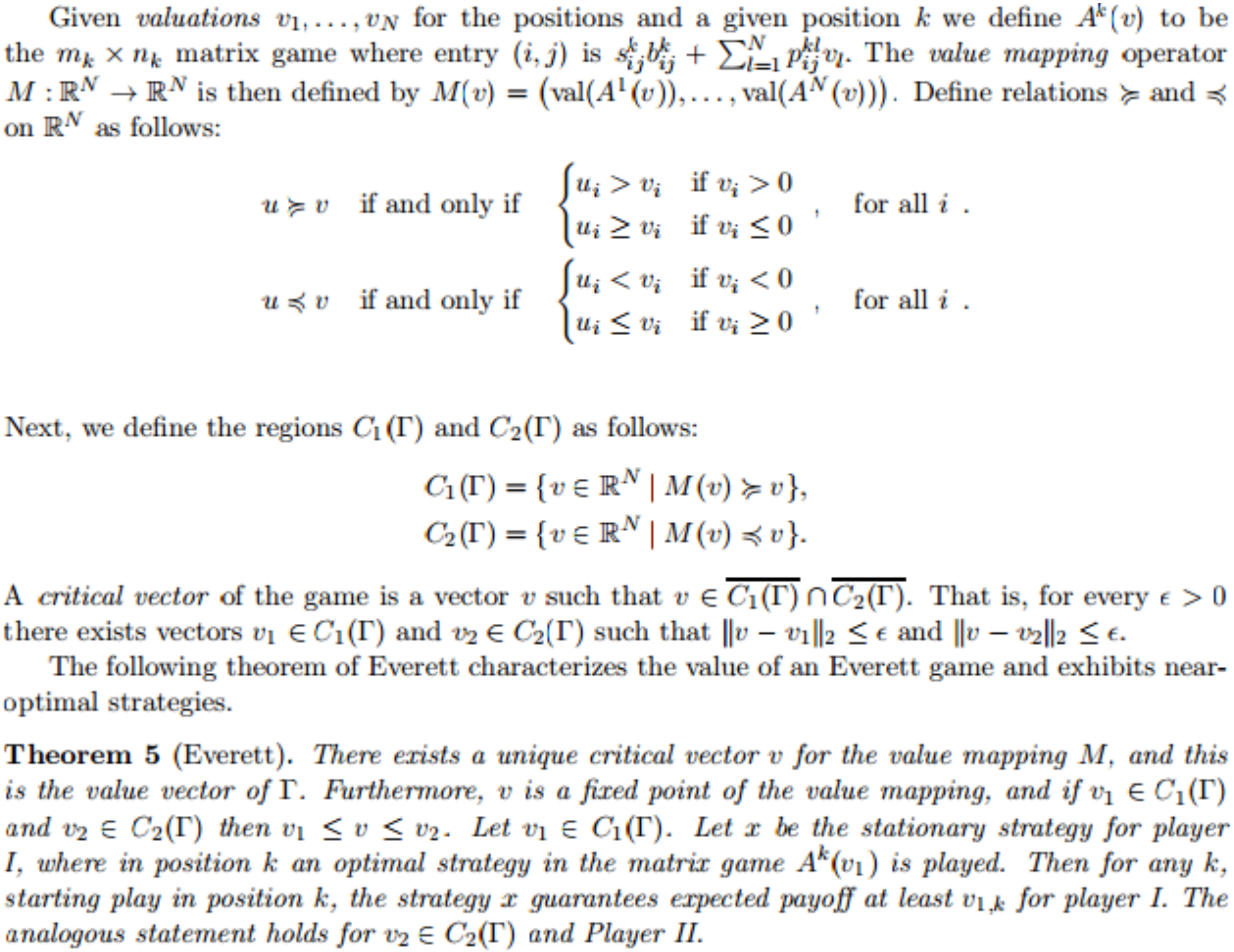
Step 2: Reduction to bounding *patience*

- We need to upper bound the difference in value between ***time bounded*** and ***infinite*** versions of the game.
- The difference in value between a time bounded and the infinite version of a concurrent reachability game is captured by the ***patience*** of its stationary near-optimal strategies.
  - Patience = 1/smallest non-zero probability used

- **Lemma**: If the game has an $\mathcal{E}$-optimal strategy with patience L, then for $T=kNL\uparrow N$, the value of the game with time bound T differs from the value of the original game by at most $\varepsilon+e\uparrow-k$.

Step 3: Bounding patience using R.A.G.

- Everett's characterization (1957) of value and near-optimal strategies:



Given *valuations* $v_1, \ldots, v_N$ for the positions and a given position k we define $A^k(v)$ to be the $m_k \times n_k$ matrix game where entry (i,j) is $s^k_{ij} b^k_{ij} + \sum_{l=1}^{N} p^{kl}_{ij} v_l$. The *value mapping* operator $M : \mathbb{R}^N \to \mathbb{R}^N$ is then defined by $M(v) = \big(\text{val}(A^1(v)), \ldots, \text{val}(A^N(v))\big)$. Define relations $\succcurlyeq$ and $\preccurlyeq$ on $\mathbb{R}^N$ as follows:

$$u \succcurlyeq v \quad \text{if and only if} \quad \begin{cases} u_i > v_i & \text{if } v_i > 0 \\ u_i \geq v_i & \text{if } v_i \leq 0 \end{cases}, \quad \text{for all } i \ .$$

$$u \preccurlyeq v \quad \text{if and only if} \quad \begin{cases} u_i < v_i & \text{if } v_i < 0 \\ u_i \leq v_i & \text{if } v_i \geq 0 \end{cases}, \quad \text{for all } i \ .$$

Next, we define the regions $C_1(\Gamma)$ and $C_2(\Gamma)$ as follows:

$$C_1(\Gamma) = \{v \in \mathbb{R}^N \mid M(v) \succcurlyeq v\},$$
$$C_2(\Gamma) = \{v \in \mathbb{R}^N \mid M(v) \preccurlyeq v\}.$$

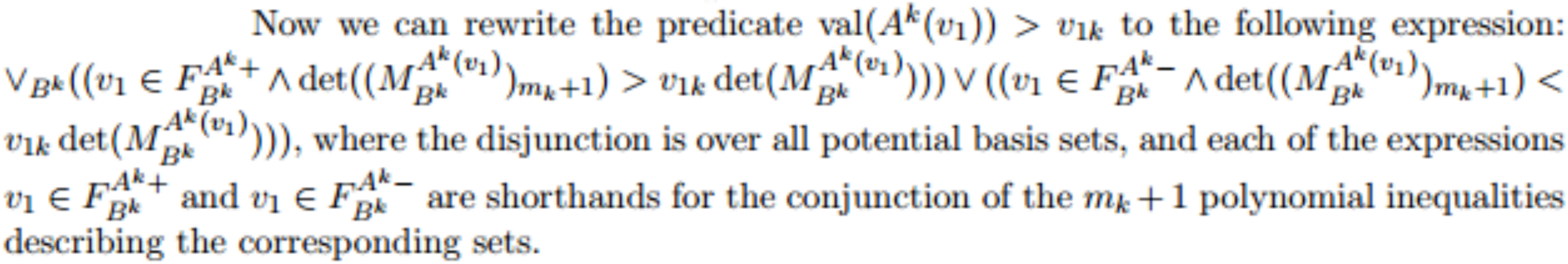
A *critical vector* of the game is a vector v such that $v \in \overline{C_1(\Gamma)} \cap \overline{C_2(\Gamma)}$. That is, for every $\epsilon > 0$ there exists vectors $v_1 \in C_1(\Gamma)$ and $v_2 \in C_2(\Gamma)$ such that $\|v - v_1\|_2 \leq \epsilon$ and $\|v - v_2\|_2 \leq \epsilon$.

The following theorem of Everett characterizes the value of an Everett game and exhibits near-optimal strategies.

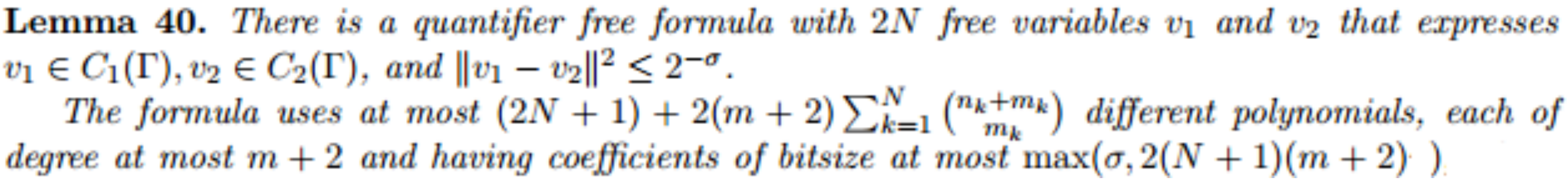
Theorem 5 (Everett). *There exists a unique critical vector v for the value mapping M, and this is the value vector of Γ. Furthermore, v is a fixed point of the value mapping, and if $v_1 \in C_1(\Gamma)$ and $v_2 \in C_2(\Gamma)$ then $v_1 \leq v \leq v_2$. Let $v_1 \in C_1(\Gamma)$. Let x be the stationary strategy for player I, where in position k an optimal strategy in the matrix game $A^k(v_1)$ is played. Then for any k, starting play in position k, the strategy x guarantees expected payoff at least $v_{1,k}$ for player I. The analogous statement holds for $v_2 \in C_2(\Gamma)$ and Player II.*

Step 3: Bounding patience using R.A.G.

- Applying the fundamental theorem of linear programming and Cramer's rule:



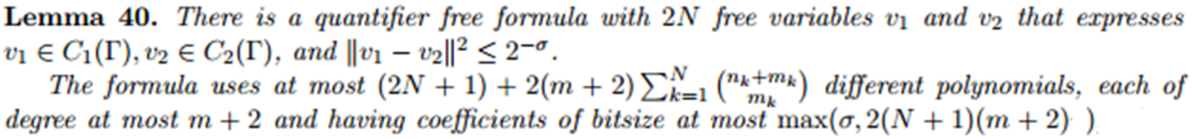
Now we can rewrite the predicate $\mathrm{val}(A^k(v_1)) > v_{1k}$ to the following expression: $\vee_{B^k}((v_1 \in F_{B^k}^{A^k+} \wedge \det((M_{B^k}^{A^k(v_1)})_{m_k+1}) > v_{1k} \det(M_{B^k}^{A^k(v_1)}))) \vee ((v_1 \in F_{B^k}^{A^k-} \wedge \det((M_{B^k}^{A^k(v_1)})_{m_k+1}) < v_{1k} \det(M_{B^k}^{A^k(v_1)}))),$ where the disjunction is over all potential basis sets, and each of the expressions $v_1 \in F_{B^k}^{A^k+}$ and $v_1 \in F_{B^k}^{A^k-}$ are shorthands for the conjunction of the $m_k + 1$ polynomial inequalities describing the corresponding sets.



Lemma 40. *There is a quantifier free formula with $2N$ free variables v_1 and v_2 that expresses $v_1 \in C_1(\Gamma), v_2 \in C_2(\Gamma)$, and $\|v_1 - v_2\|^2 \leq 2^{-\sigma}$.*

The formula uses at most $(2N+1) + 2(m+2)\sum_{k=1}^{N} \binom{n_k+m_k}{m_k}$ different polynomials, each of degree at most $m+2$ and having coefficients of bitsize at most $\max(\sigma, 2(N+1)(m+2))$

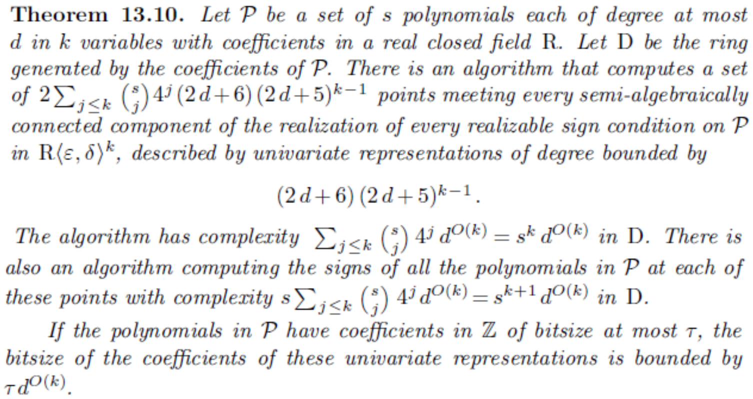
Step 3: Bounding patience using R.A.G.



Lemma 40. *There is a quantifier free formula with $2N$ free variables v_1 and v_2 that expresses $v_1 \in C_1(\Gamma), v_2 \in C_2(\Gamma)$, and $\|v_1 - v_2\|^2 \leq 2^{-\sigma}$.*

The formula uses at most $(2N+1) + 2(m+2)\sum_{k=1}^{N} \binom{n_k+m_k}{m_k}$ different polynomials, each of degree at most $m+2$ and having coefficients of bitsize at most $\max(\sigma, 2(N+1)(m+2)\ )$.

+



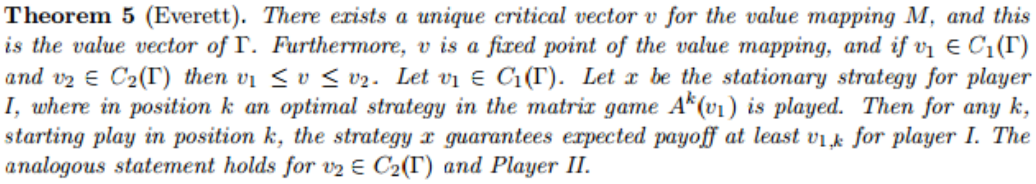
Theorem 13.10. *Let $\mathcal{P}$ be a set of s polynomials each of degree at most d in k variables with coefficients in a real closed field R. Let D be the ring generated by the coefficients of $\mathcal{P}$. There is an algorithm that computes a set of $2\sum_{j \leq k} \binom{s}{j} 4^j (2\,d+6)\,(2\,d+5)^{k-1}$ points meeting every semi-algebraically connected component of the realization of every realizable sign condition on $\mathcal{P}$ in $\mathrm{R}\langle\varepsilon,\delta\rangle^k$, described by univariate representations of degree bounded by*

$$(2\,d+6)\,(2\,d+5)^{k-1}.$$

The algorithm has complexity $\sum_{j \leq k} \binom{s}{j} 4^j\, d^{O(k)} = s^k\, d^{O(k)}$ in D. There is also an algorithm computing the signs of all the polynomials in $\mathcal{P}$ at each of these points with complexity $s\sum_{j \leq k} \binom{s}{j} 4^j\, d^{O(k)} = s^{k+1}\, d^{O(k)}$ in D.

If the polynomials in $\mathcal{P}$ have coefficients in $\mathbb{Z}$ of bitsize at most τ, the bitsize of the coefficients of these univariate representations is bounded by $\tau d^{O(k)}$.

+



Theorem 5 (Everett). *There exists a unique critical vector v for the value mapping M, and this is the value vector of Γ. Furthermore, v is a fixed point of the value mapping, and if $v_1 \in C_1(\Gamma)$ and $v_2 \in C_2(\Gamma)$ then $v_1 \leq v \leq v_2$. Let $v_1 \in C_1(\Gamma)$. Let x be the stationary strategy for player I, where in position k an optimal strategy in the matrix game $A^k(v_1)$ is played. Then for any k, starting play in position k, the strategy x guarantees expected payoff at least $v_{1,k}$ for player I. The analogous statement holds for $v_2 \in C_2(\Gamma)$ and Player II.*

\+ separation bounds for roots of univariate polynomials (Cauchy)

=

An ε-optimal strategy with all probabilities either 0 or bounded from below by $\varepsilon \uparrow m \uparrow O(N)$

Main theorem

For all games with *N* positions and *m* actions for each player in each position, $(1/\varepsilon)^{m^{O(N)}}$ iterations is sufficient to arrive at ε-optimal strategy.

Tight example

Generalized Purgatory P(*N*,*m*):

- Column player repeatedly hides a number in {1,..,*m*}.
- Row player must try to guess the number.
- If he guesses correctly *N* times in a row, he wins the game.
- If he ever guesses incorrectly ***overshooting*** hidden number, he loses the game.

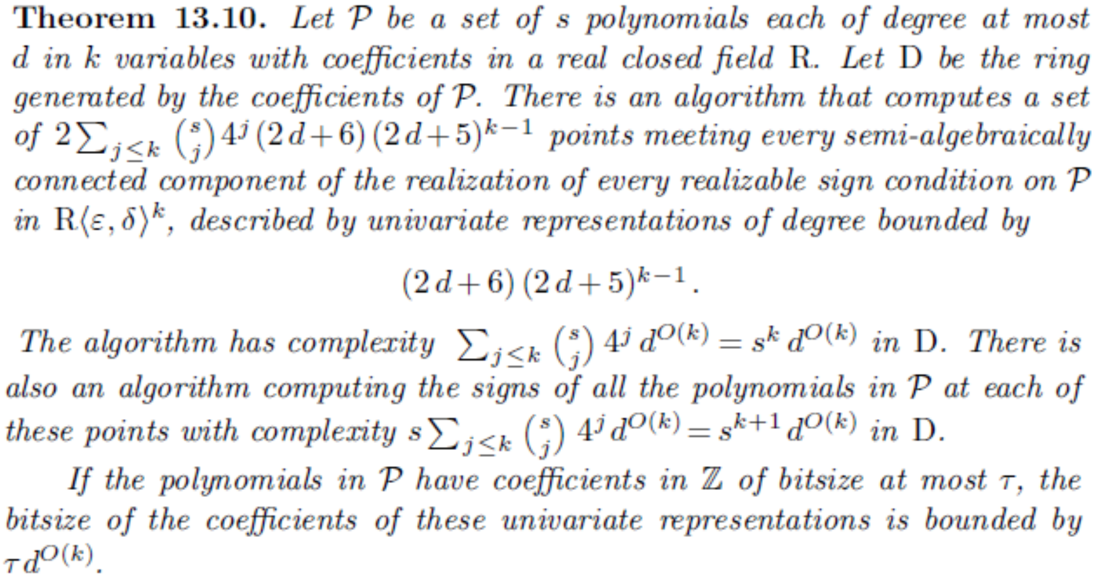
– These games all have value 1(!)

– Strategy iteration needs $(1/\varepsilon)^{m^{N\text{-o}(N)}}$ to get ε-optimal strategy.

Main theorem

For all games with N positions and m actions for each player in each position, $(1/\varepsilon)^{m^{O(N)}}$ iterations is sufficient to arrive at ε-optimal strategy.

R.A.G. engine: The sampling Theorem

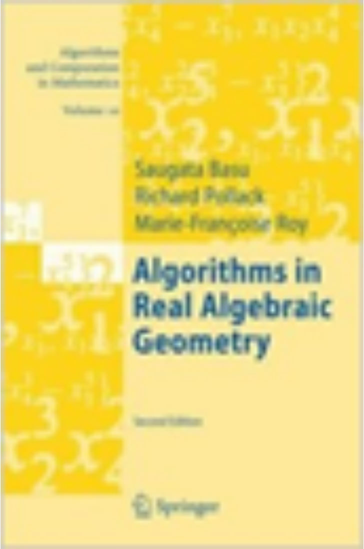


Theorem 13.10. *Let $\mathcal{P}$ be a set of s polynomials each of degree at most d in k variables with coefficients in a real closed field* R. *Let* D *be the ring generated by the coefficients of $\mathcal{P}$. There is an algorithm that computes a set of $2\sum_{j\le k}\binom{s}{j}4^j(2d+6)(2d+5)^{k-1}$ points meeting every semi-algebraically connected component of the realization of every realizable sign condition on $\mathcal{P}$ in $\mathrm{R}\langle\varepsilon,\delta\rangle^k$, described by univariate representations of degree bounded by*

$$(2d+6)(2d+5)^{k-1}.$$

The algorithm has complexity $\sum_{j\le k}\binom{s}{j}4^j d^{O(k)} = s^k d^{O(k)}$ in D. *There is also an algorithm computing the signs of all the polynomials in $\mathcal{P}$ at each of these points with complexity $s\sum_{j\le k}\binom{s}{j}4^j d^{O(k)} = s^{k+1} d^{O(k)}$ in* D.

If the polynomials in $\mathcal{P}$ have coefficients in $\mathbb{Z}$ of bitsize at most τ, the bitsize of the coefficients of these univariate representations is bounded by $\tau d^{O(k)}$.



Saugata Basu
Richard Pollack
Marie-Françoise Roy

Algorithms in Real Algebraic Geometry

Springer

Thank you!