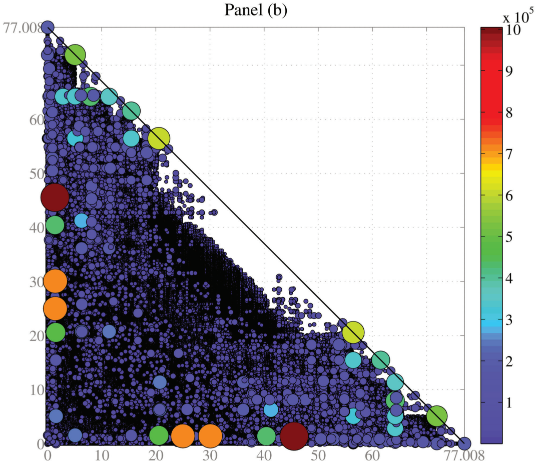
Structural Estimation of Dynamic Directional Games with Multiple Equilibria

Fedor Iskhakov, Research School of Economics, ANU
Dennis Kristensen, University College London
John Rust, Georgetown University
Bertel Schjerning, University of Copenhagen

RSE Seminar

Australian National University, November 29, 2023

Multiplicity of equilibria



Color and size of dots denote number of repetitions of the same payoff

Structural estimation of stochastic dynamic games

- Structurally estimation of stochastic dynamic discrete games
  - Several decision makers (*players*) makes discrete choices
  - Maximize discounted expected utility/profit
  - Anticipate consequences of their current actions
  - Anticipate actions by other players in current and future periods
  - Equilibrium concept: MPE
- Problem: Multiplicity of equilibria
  - NFXP-MLE needs to repeatedly solve for all MPE
    for every evaluation of the likelihood function
  - MPEC-MLE avoids repeated solution of all MPE,
    but suffer from serious issues with local minima.
  - Two-step (CCP) methods avoids full solution,
    but is inefficient and suffer from small sample bias
  - Sequential methods (NPL/EPL) should fix this, but can be unstable.
- This paper: Nested recursive lexicographical search (NRLS)
  - NFXP-type MLE estimator that avoids full enumeration
  - Builds on full solution methods for directional dynamic games
  - Compare to existing estimators (two-step, NPL, EPL, MPEC)

Markov Perfect Equilibria

- MPE is a pair of strategy profile and value functions
- In compact notation

$$\begin{aligned} \mathsf{V} &= \Psi^{\mathsf{V}}(\mathsf{V}, \mathsf{P}, \theta) \\ \mathsf{P} &= \Psi^{\mathsf{P}}(\mathsf{V}, \mathsf{P}, \theta) \end{aligned}$$

- Set of all Markov Perfect Equilibria

$$SOL(\Psi, \theta) = \left\{ (\mathsf{P}, \mathsf{V}) \;\middle|\; \begin{array}{l} \mathsf{V} = \Psi^{\mathsf{V}}(\mathsf{V}, \mathsf{P}, \theta) \\ \mathsf{P} = \Psi^{\mathsf{P}}(\mathsf{V}, \mathsf{P}, \theta) \end{array} \right\}$$

- $\Psi^{\mathsf{V}}: \ \mathsf{V}, \mathsf{P} \longrightarrow \mathsf{V}$ Bellman Optimality
- $\Psi^{\mathsf{P}}: \ \mathsf{V}, \mathsf{P} \longrightarrow \mathsf{P}$ Bayes-Nash Equilibrium (logit CCPs)
- $\Gamma: \ \mathsf{P} \longrightarrow \mathsf{V}$ Hotz-Miller inversion

Maximum Likelihood Estimation

- Data from M independent markets from T periods
  $\mathrm{Z} = \{\bar{\mathrm{a}}^{mt}, \bar{\mathrm{x}}^{mt}\}_{m \in \mathcal{M}, t \in \mathcal{T}}$
  Usually assume only one equilibrium is played in the data.
- For a given θ, let
  $\left(\mathrm{P}^{\ell}(\theta), \mathrm{V}^{\ell}(\theta)\right) \in SOL(\Psi, \theta)$ denote the ℓ-the equilibrium
- Log-likelihood function is

$$\mathcal{L}(Z, \theta) = \max_{(\mathrm{P}^{\ell}(\theta), \mathrm{V}^{\ell}(\theta) \in SOL(\Psi, \theta)} \frac{1}{M} \sum_{i=1}^{N} \sum_{m=1}^{M} \sum_{t=1}^{T} \log P_i^{\ell}(\bar{a}_i{}^{mt} | \bar{\mathrm{x}}^{mt}; \theta)$$

- The ML estimator is $\theta^{ML} = \arg\max_{\theta} \mathcal{L}(Z, \theta)$

Estimation methods for *dynamic* stochastic games

- Two step (CCP) estimators
  - Fast, potentially large finite sample biases

Hotz, Miller (1993); Altug, Miller (1998); Pakes, Ostrovsky, and Berry (2007); Pesendorfer, Schmidt-Dengler (2008)

1. Estimate CCP $\rightarrow \hat{\mathsf{P}}$
2. Method of moments • Minimal distance • Pseudo likelihood

$$\min_{\theta} \left[\hat{\mathsf{P}} - \Psi^{\mathsf{P}}(\Gamma(\theta, \hat{\mathsf{P}}), \hat{\mathsf{P}}, \theta)\right]' W \left[\hat{\mathsf{P}} - \Psi^{\mathsf{P}}(\Gamma(\theta, \hat{\mathsf{P}}), \hat{\mathsf{P}}, \theta)\right]$$

$$\max_{\theta} \mathcal{L}(\mathsf{Z}, \Psi^{\mathsf{P}}(\Gamma(\theta, \hat{\mathsf{P}}), \hat{\mathsf{P}}, \theta))$$

- Nested pseudo-likelihood (NPL)
  - Recursive two step pseudo-likelihood
  - Bridges the gap between efficiency and tractability
  - Unstable under multiplicity

Aguirregabiria, Mira (2007); Pesendorfer, Schmidt-Dengler (2010); Kasahara and Shimotsu (2012); Aguirregabiria, Marcoux (2021)

Estimation methods for *dynamic* stochastic games

- Efficient and Convergent Sequential Pseudo-Likelihood Estimation of Dynamic Discrete Games (EPL)
- NPL-inspired estimator, EPL, that (according to authors):
  1. Retains and improves on advantages of NPL in games.
     - Addresses multiple equilibria via two-step estimation.
     - Avoids repeatedly solving for all equilibria.
     - Exploits natural structure of the model for estimation.
     - Simple estimation for structural parameters.
  2. Extends single-agent properties of NPL to dynamic games.
     - Convergence, Efficiency, Linearity
  3. Works well in difficult example models.

Blevins and Dearing, ReStud (forthcoming)

Estimation methods for *dynamic* stochastic games

- Equilibrium inequalities (BBL)
  - Minimize the one-sided discrepancies
  - Computationally feasible in large models

  Bajari, Benkard, Levin (2007)

- Math programming with equilibrium constraints (MPEC)
  - MLE as constrained optimization
  - Does not rely on the structure of the problem
  - Much bigger computational problem

  Su (2013); Egesdal, Lai and Su (2015)

$$\max_{(\theta, \mathsf{P}, \mathsf{V})} \mathcal{L}(\mathsf{Z}, \mathsf{P}) \text{ subject to } \mathsf{V} = \Psi^{\mathsf{V}}(\mathsf{V}, \mathsf{P}, \theta), \mathsf{P} = \Psi^{\mathsf{P}}(\mathsf{V}, \mathsf{P}, \theta)$$

- All solution homotopy MLE

  Borkovsky, Doraszelsky and Kryukov (2010)

Overview of NRLS

- Robust and *computationally feasible*$^{(?)}$ MLE estimator for directional dynamic games (DDG)
- Rely of full solution algorithm that provably computes all MPE under certain regularity conditions
- Employ discrete programming method (BnB) to maximize likelihood function over the finite set of equilibria
- Use non-parametric likelihood to refine BnB algorithm

- Fully robust to multiplicity of MPE
- Relax single-equilibrium-in-data assumption
- Path to estimation of equilibrium selection rules
- Avoids full enumeration in larger samples

ROAD MAP

1. Solving directional dynamic games (DDGs):
   - Simple example: Bertrand pricing and investment game
   - State recursion algorithm
   - Recursive lexicographical search (RLS) algorithm
2. Structural estimation of DDGs using Nested RLS
3. Refinements of NRLS: The need for speed
4. Monte Carlo: (Compare NRLS, two-step CCP, NPL, EPL, MPEC)

Dynamic Bertrand price competition

Directional stochastic dynamic game

- Two Bertrand competitors, $n = 2$, no entry or exit
- Discrete time, infinite horizon ($t = 1, 2, \ldots, \infty$)
- Firms maximize expected discounted profits
- Each firm has two choices in each period:
  1. Price for the product — simultaneous
  2. Whether or not to buy the state of the art technology
     - Simultaneous moves
     - Alternating moves

Static Bertrand price competition in each period

- Continuum of consumers make static purchase decision
- No switching costs: buy from the lower price supplier
- Per period profits (c_i is the marginal cost)

$$r_i(c_1, c_2) = \begin{cases} 0 & \text{if } c_i \geq c_j \\ c_j - c_i & \text{if } c_i < c_j \end{cases}$$

Cost-reducing investments

State-of-the-art production cost c process

- Initial value c_0, lowest value 0: $0 \leq c \leq c_0$
- Discretized with n points
- Follows exogenous Markov process and only improves
- Markov transition probability $\pi(c_{t+1}|c_t)$
  $\pi(c_{t+1}|c_t) = 0$ if $c_{t+1} > c_t$

State space of the problem

- State of the game: cost structure (c_1, c_2, c)
- State space is $S = (c_1, c_2, c) \subset R^3$: $c_1 \geq c$, $c_2 \geq c$
- Actions are observable
- Private information EV(1) i.i.d. shocks $\eta\epsilon_{i,I}$ and $\eta\epsilon_{i,N}$

Definition of Markov Perfect Equilibium

Definition (Markov perfect equilibrium (MPE))

MPE of Bertrand investment stochastic game is a pair of

- strategy profile $\sigma^* = (\sigma_1^*, \sigma_2^*)$, and
- pair of *value functions* $V(s) = \Big(V_1(s), V_2(s)\Big)$, $V_i : S \to R$,

such that

1. Bellman equations (below) are satisfied for each firm, and
2. strategies σ_1^* and σ_2^* constitute mutual best responses, and assign positive probabilities only to the actions in the set of maximizers of the Bellman equations.

Bellman equations, firm $i = 1$, simultaneous moves

$$V_i(c_1, c_2, c) = \max\left[v_i^I(c_1, c_2, c) + \eta\epsilon_{i,I}, v_i^N(c_1, c_2, c) + \eta\epsilon_{i,N}\right]$$

$$\begin{aligned} v_i^N(c_1, c_2, c) &= r^i(c_1, c_2) + \beta EV_i(c_1, c_2, c|N) \\ v_i^I(c_1, c_2, c) &= r^i(c_1, c_2) - K(c) + \beta EV_i(c_1, c_2, c|I) \end{aligned}$$

With extreme value shocks, the investment probability is

$$P_i^I(c_1, c_2, c) = \frac{\exp\{v_I^I(c_1, c_2, c)/\eta\}}{\exp\{v_i^I(c_1, c_2, c)/\eta\} + \exp\{v_i^N(c_1, c_2, c)/\eta\}}$$

Bellman equations, firm $i = 1$, simultaneous moves

The expected values are given by

$$\begin{aligned} EV_i(c_1, c_2, c|N) &= \int_0^c \left[P_j^I(c_1, c_2, c) H_i(c_1, c, c') + \right. \\ & \left. P_j^N(c_1, c_2, c) H_i(c_1, c_2, c') \right] \pi(dc'|c) \\ EV_i(c_1, c_2, c|I) &= \int_0^c \left[P_j^I(c_1, c_2, c) H_i(c, c, c') + \right. \\ & \left. P_j^N(c_1, c_2, c) H_i(c, c_2, c') \right] \pi(dc'|c) \end{aligned}$$

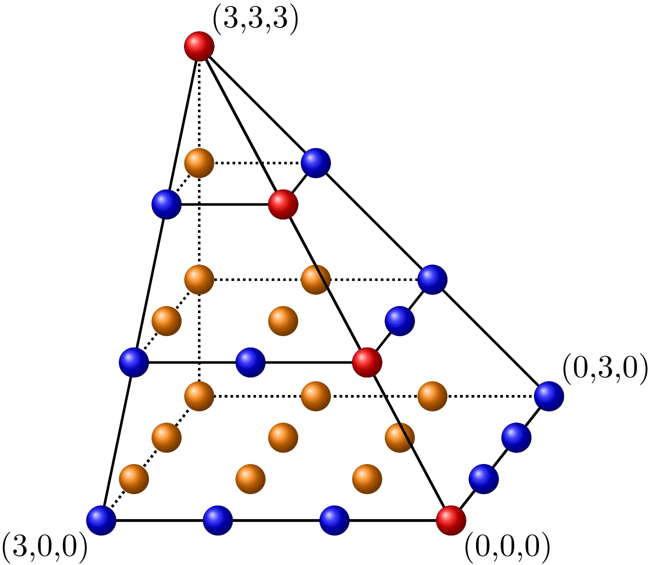
where

$$H_i(c_1, c_2, c) = \eta \log \left[\exp\left(v_i^N(c_1, c_2, c)/\eta\right) + \exp\left(v_i^I(c_1, c_2, c)/\eta\right) \right]$$

is the "smoothed max" or logsum function

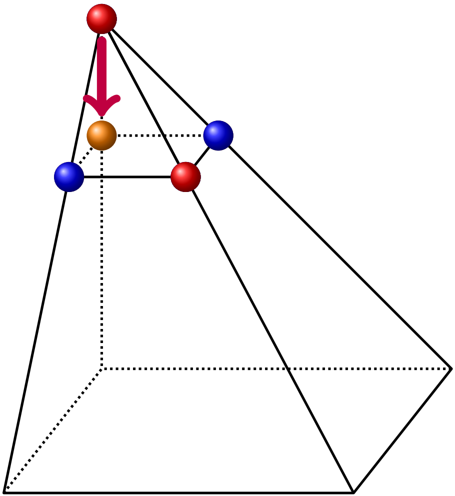
Discretized state space = a "quarter pyramid"

$S = \{(c_1, c_2, c) | c_1 \geq c,\ c_2 \geq c, c \in [0, 3]\},\ n = 4$



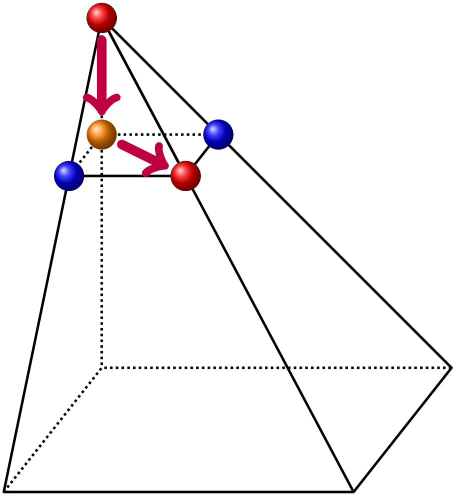
Game dynamics: example

The game starts at the apex, as some point technology improves



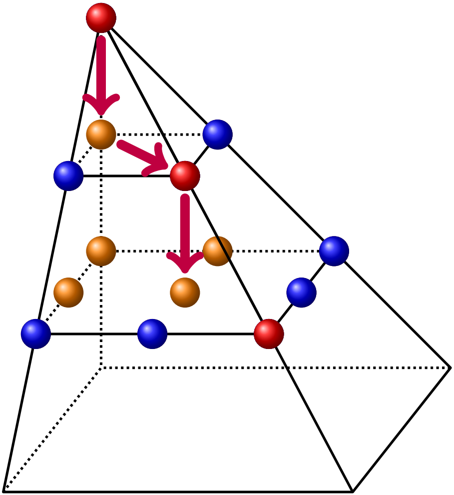
Game dynamics: example

Both firms buy new technology $c = 2 \rightsquigarrow (c_1, c_2, c) = (2, 2, 2)$



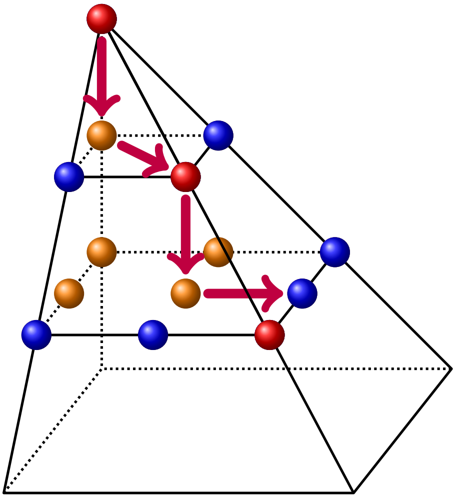
Game dynamics: example

State-of-the-art technology becomes $c = 1 \rightsquigarrow (c_1, c_2, c) = (2, 2, 1)$



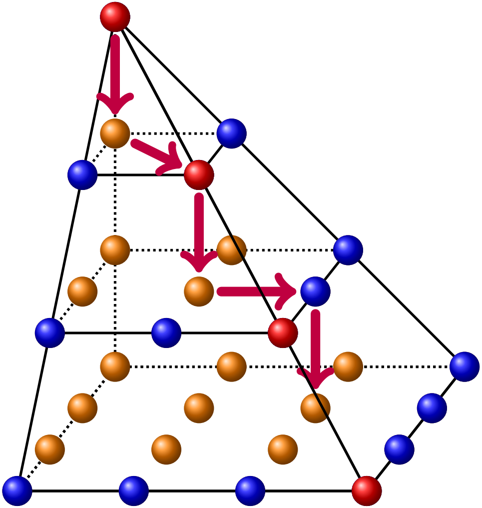
Game dynamics: example

Firm 1 invests and becomes cost leader $\rightsquigarrow (c_1, c_2, c) = (1, 2, 1)$



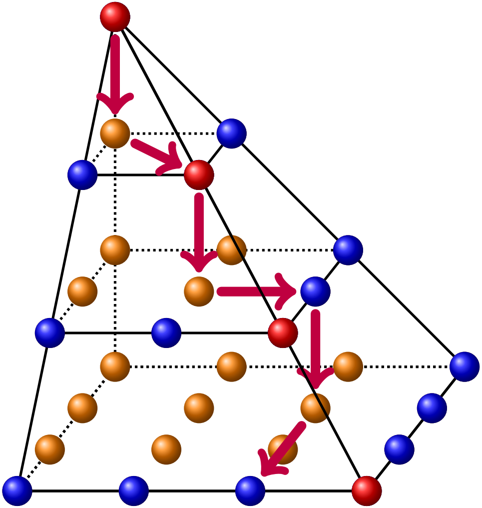
Game dynamics: example

State-of-the-art technology becomes $c = 0 \rightsquigarrow (c_1, c_2, c) = (1, 2, 0)$



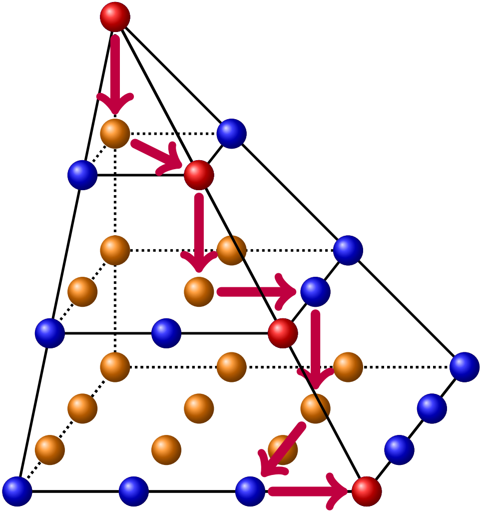
Game dynamics: example

Firm 2 leapfrogs firm 1 to become new cost leader $\rightsquigarrow$ $(c_1, c_2, c) = (1, 0, 0)$



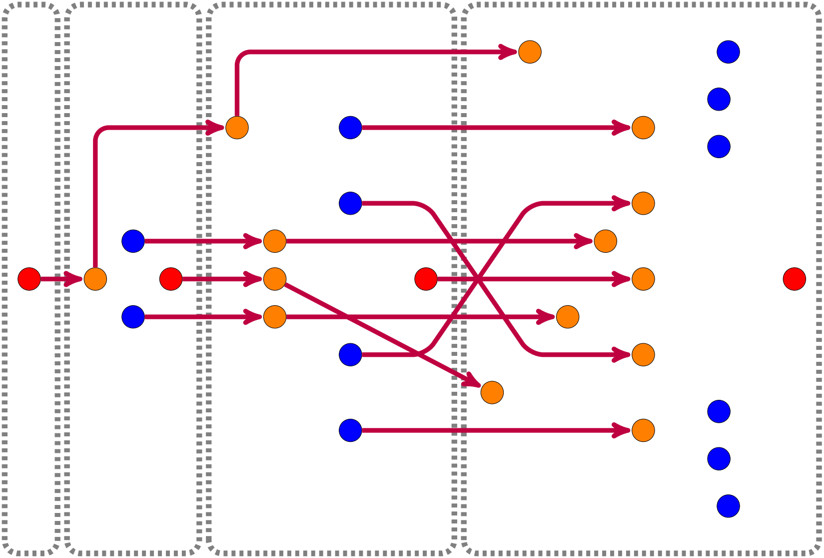
Game dynamics: example

Firm 1 invests, and the game reaches terminal state $\rightsquigarrow$ $(c_1, c_2, c) = (0, 0, 0)$



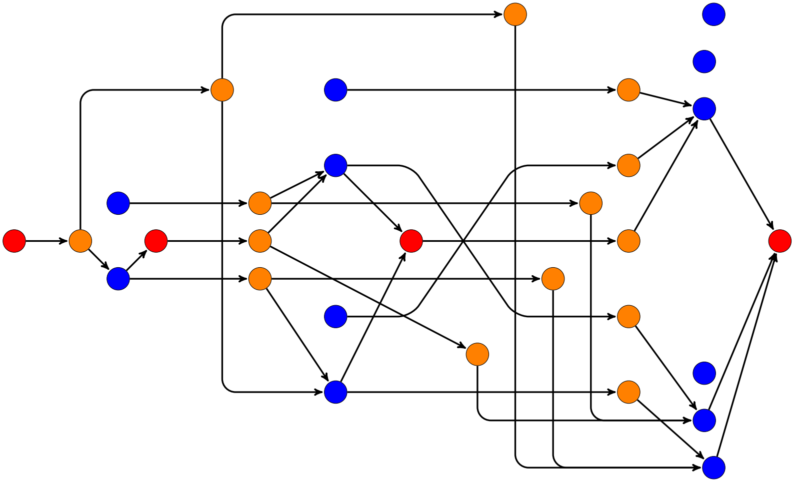
Transitions due to technological progress

As c decreases, the game falls through the layers of the pyramid



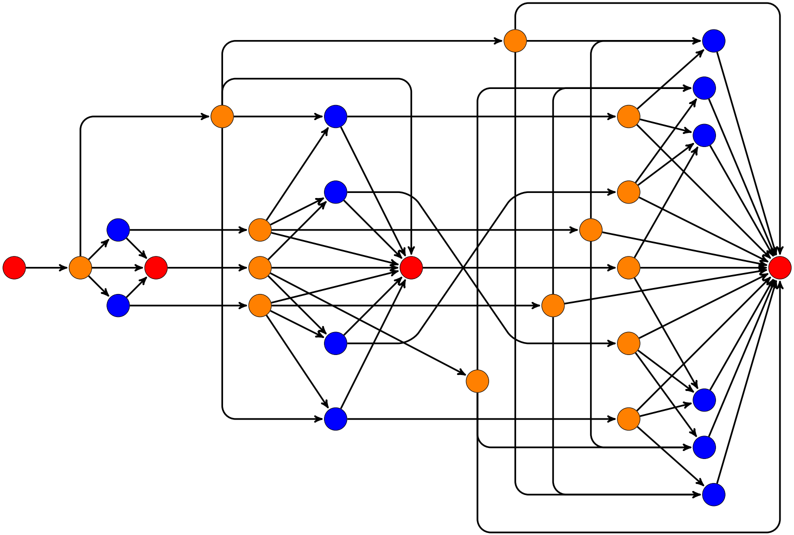
Strategy-specific partial order on S

Strategy $\sigma = (\sigma_1, \sigma_2)$ of both firms



Strategy independent partial order on S

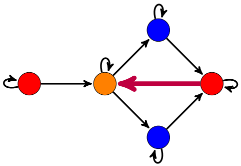
Coarsest common refinement of partial orders induced by all strategies



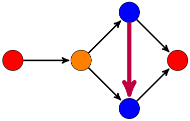
Definition of the Dynamic Directional Games

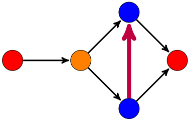
Finite state Markovian stochastic game is a DDG if it holds:

1. Every feasible Markovian strategy σ satisfies the no loop condition.



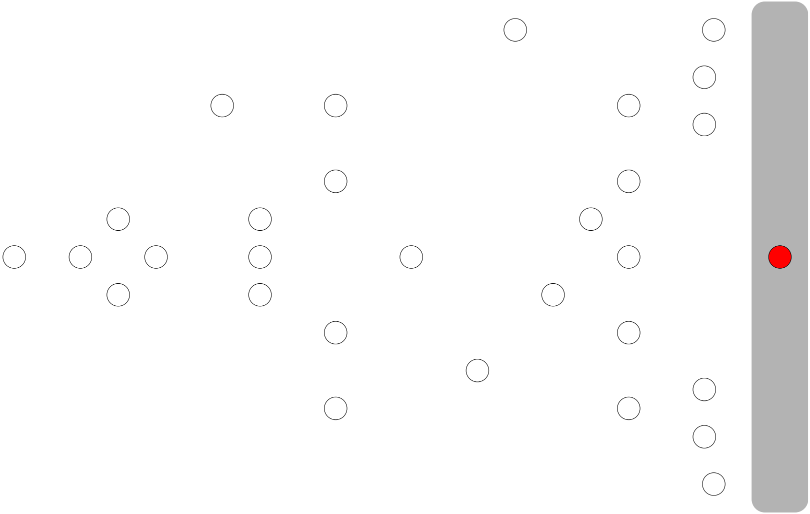
2. Every pair of feasible Markovian strategies σ and σ' induce consistent partial orders on the state space.





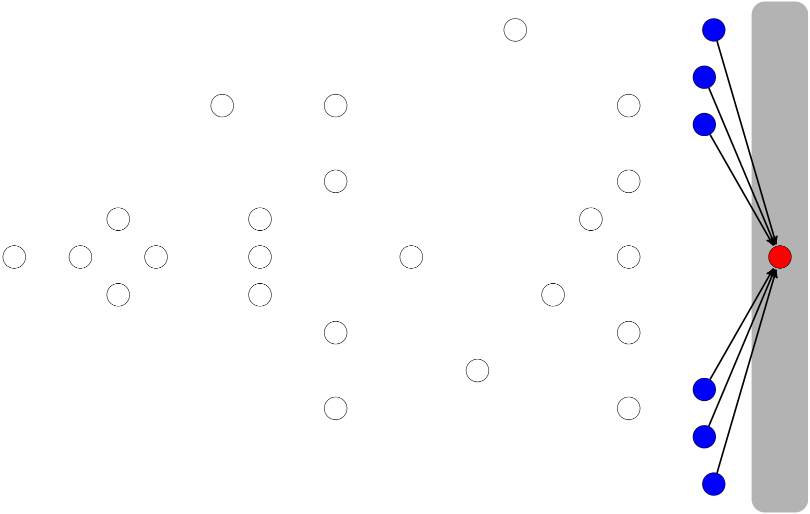
State recursion algorithm

Backward induction on stages of DDG



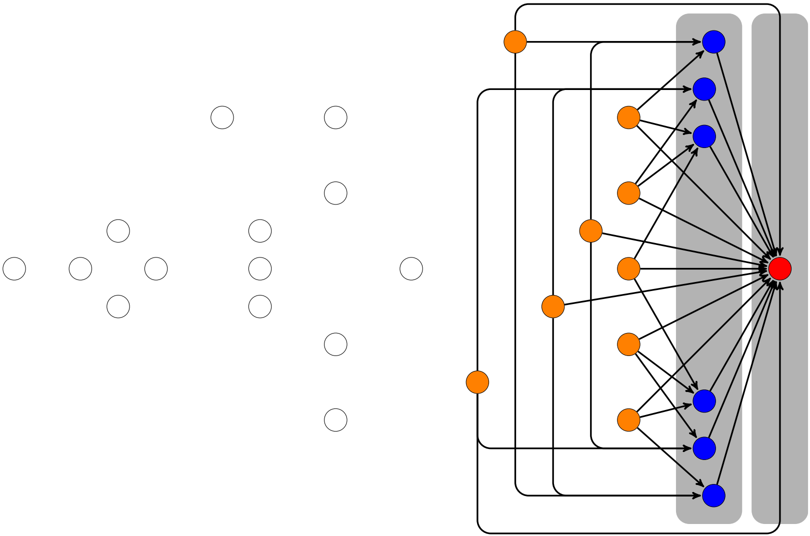
State recursion algorithm

Backward induction on stages of DDG



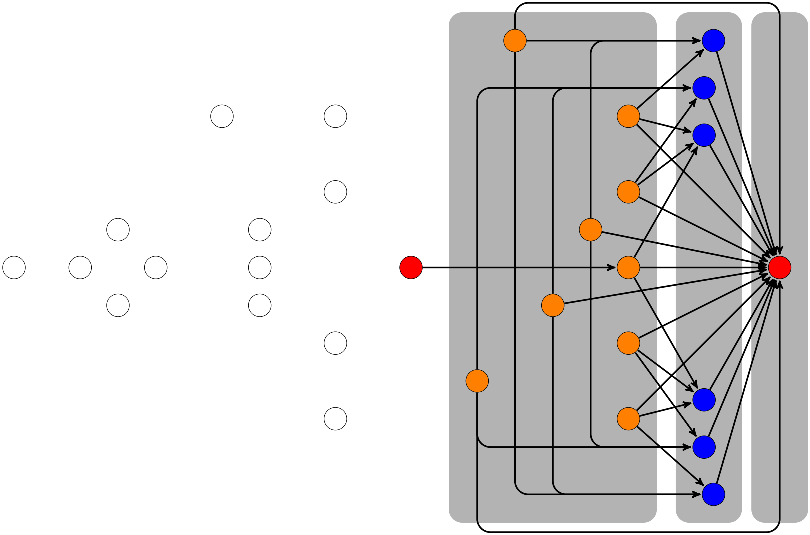
State recursion algorithm

Backward induction on stages of DDG



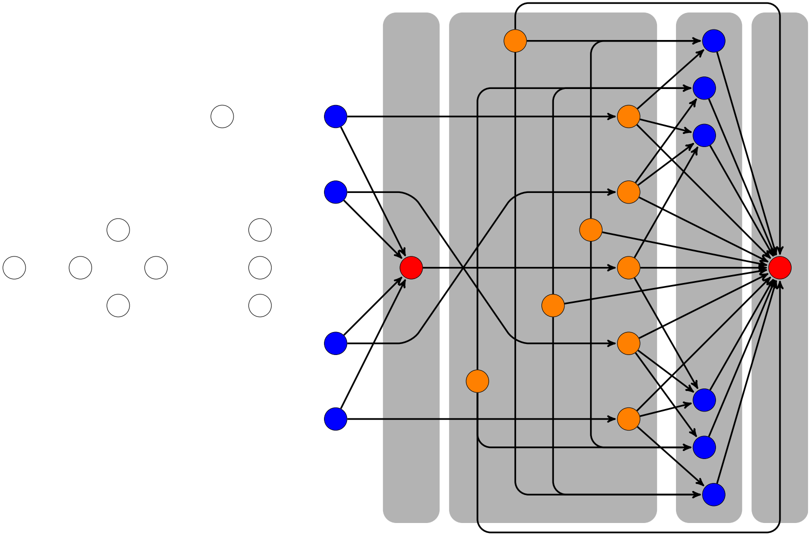
State recursion algorithm

Backward induction on stages of DDG



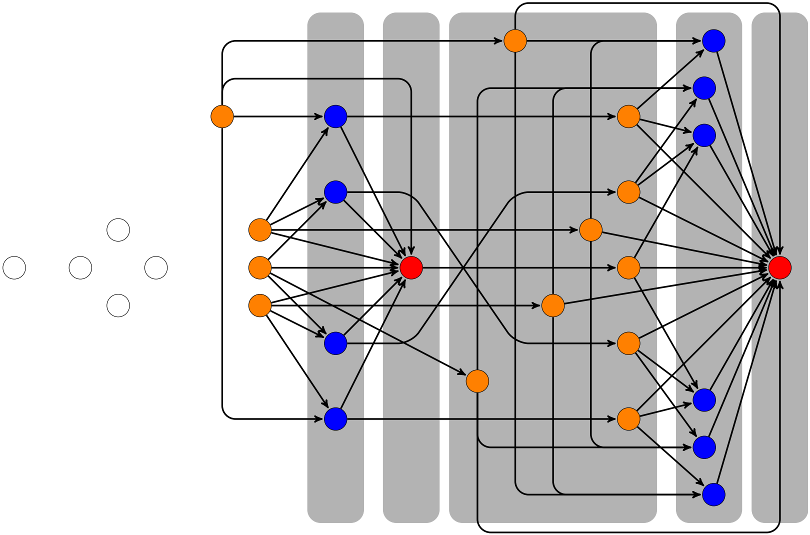
State recursion algorithm

Backward induction on stages of DDG



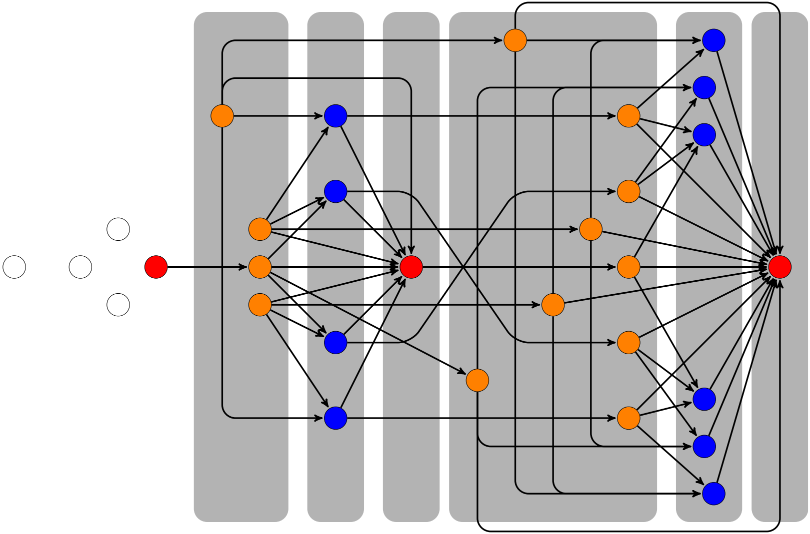
State recursion algorithm

Backward induction on stages of DDG



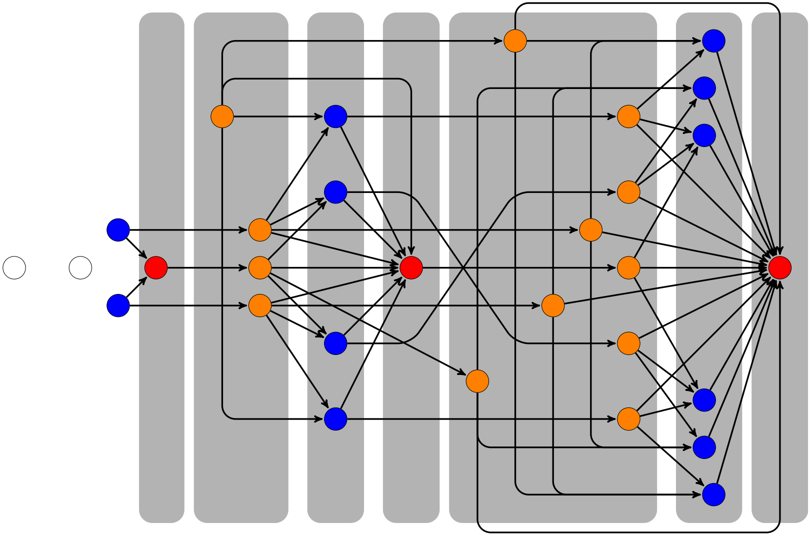
State recursion algorithm

Backward induction on stages of DDG



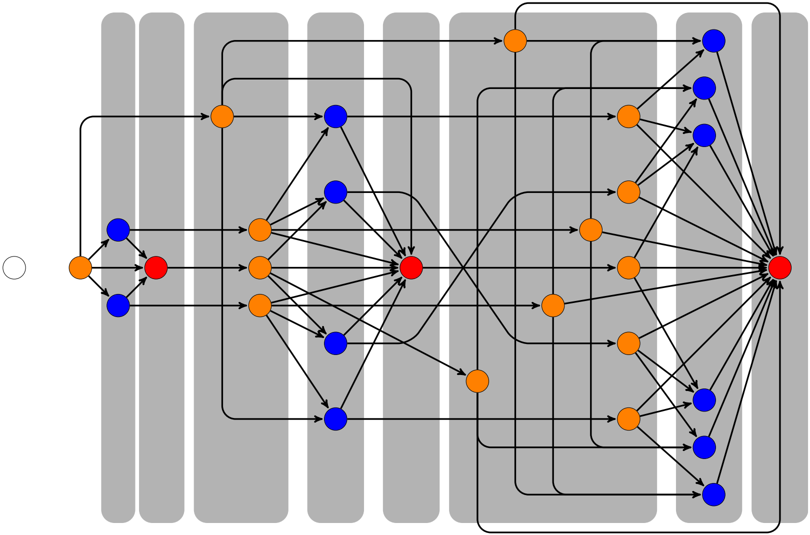
State recursion algorithm

Backward induction on stages of DDG



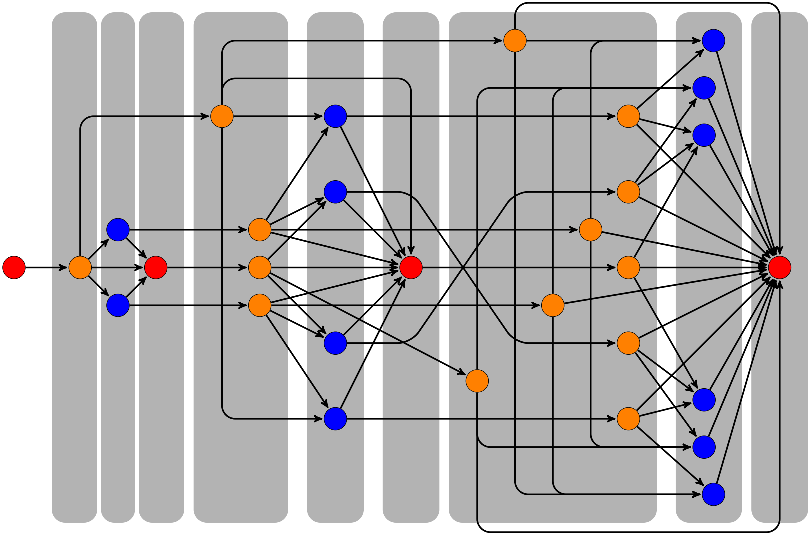
State recursion algorithm

Backward induction on stages of DDG



State recursion algorithm

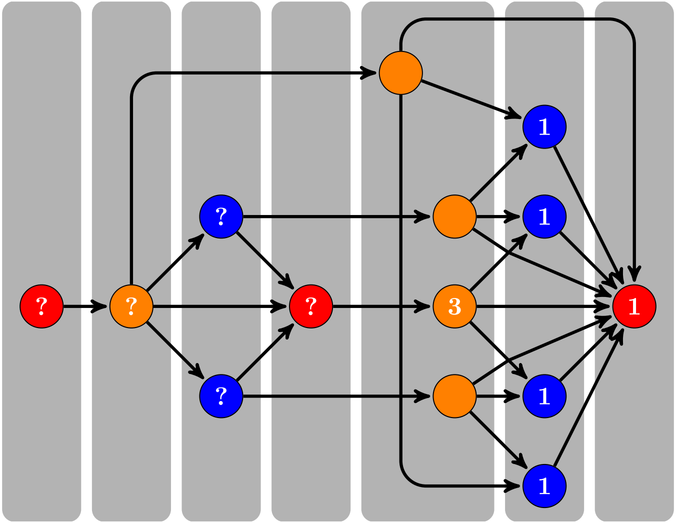
Backward induction on stages of DDG



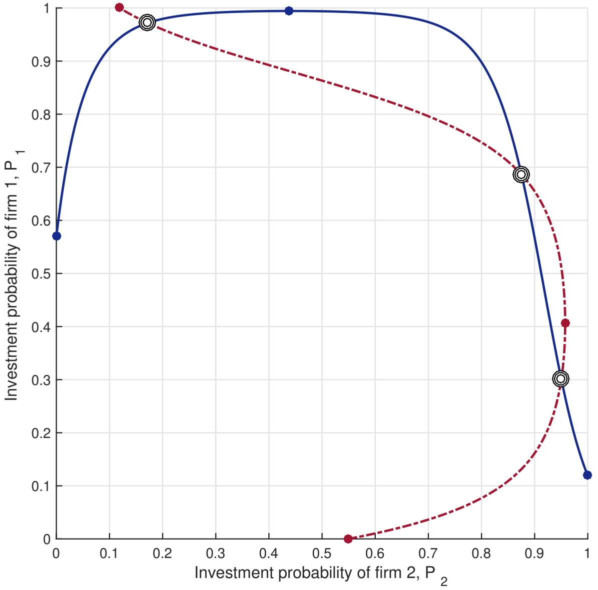
f

Multiplicity of stage equibiria

Number of equilibria in the higher stages depends on the selected equilibria



Best response functions



Recursive Lexicographic Search Algorithm

Building blocks of RLS algorithm:

1. State recursion algorithm solves the game conditional on equilibrium selection rule (ESR)
2. RLS algorithm efficiently cycles through all feasible ESRs

Challenge:

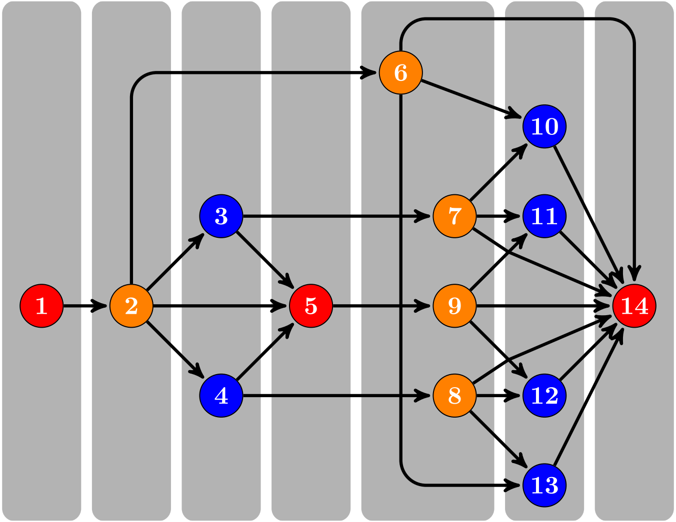
- Choice of a particular MPE for any stage game at any stage
- may alter the set and even the number of stage equilibria at earlier stages

Need to find feasible ESRs

- ESR = string of digits that index the selected stage equilibrium in each point

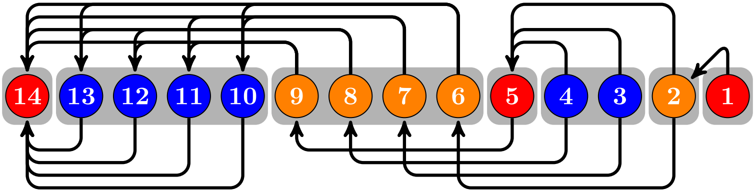
Indexing of points in the state space

Lower index for dependent points, highest for terminal stage

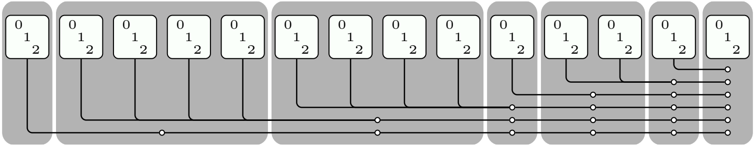


Preserving stage order in ESR strings

Formalization of the ESR as stings of digits



- Digits arranged to preserve the dependence structure



Represent ESR as string of digits

Use numbers in base-K number system with digits $0, 1, .., K-1$

Dependence preserving property:
Any point of the state space may depend on the points to the left (higher digits) and not the points to the right (lower digits)

corner

edges   interior

	c	e	e	e	e	i	i	i	i	c	e	e	i	c
ESR string	14	13	12	11	10	9	8	7	6	5	4	3	2	1
c	0	0	0	0	0	0	0	0	0	1	1	1	1	2
$c1$	0	0	0	2	1	2	2	1	1	1	1	2	2	2
$c2$	0	2	1	0	0	2	1	2	1	1	2	1	2	2

All possible ESR in lexicographic order

	c	e	e	e	e	i	i	i	i	c	e	e	i	c
ESR string	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Lexicograph	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	0	0	0	0	0	2
	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	1	1
	0	0	0	0	0	0	0	0	0	0	0	0	1	2
	0	0	0	0	0	0	0	0	0	0	0	0	2	0
	0	0	0	0	0	0	0	0	0	0	0	0	2	1
	0	0	0	0	0	0	0	0	0	0	0	0	2	2
	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	1	0	1
														…
	2	2	2	2	2	2	2	2	2	2	2	2	2	0
	2	2	2	2	2	2	2	2	2	2	2	2	2	1
	2	2	2	2	2	2	2	2	2	2	2	2	2	2

4,782,969

Recalculation of feasibility condition for new ESR

Avoid recalculation of subgames

	c	e	e	e	e	i	i	i	i	c	e	e	i	c	
ESR string	14	13	12	11	10	9	8	7	6	5	4	3	2	1	
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	always admissible
Nr of eqb	1	1	1	1	1	3	3	3	3	1	1	1	**3**	1	
	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	
	0	0	0	0	0	0	0	0	0	0	0	0	0	**2**	
	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	admissible, solve
	1	1	1	1	1	3	3	3	3	1	1	1	3	*	

No changes in the solution of the game including the number of stage equilibria

Might have change

Jumping over blocks of infeasibles ESRs

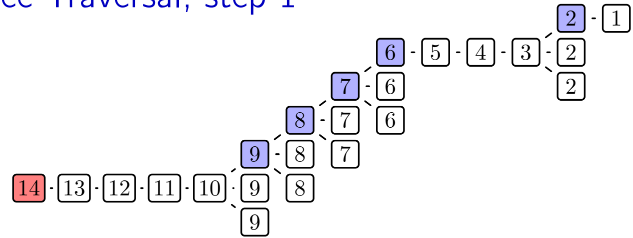
	c	e	e	e	e	i	i	i	i	c	e	e	i	c	
ESR string	14	13	12	11	10	9	8	7	6	5	4	3	2	1	Iteration:
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	1
	1	1	1	1	1	3	3	3	3	1	1	1	**3**	**1**	1a
	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	
	0	0	0	0	0	0	0	0	0	0	0	0	0	2	
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	2
	1	1	1	1	1	3	3	3	3	1	1	1	**3**	**1**	2a
	0	0	0	0	0	0	0	0	0	0	0	0	1	**1**	
	0	0	0	0	0	0	0	0	0	0	0	0	1	2	
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**2**	**0**	3
	1	1	1	1	1	3	3	3	**3**	**1**	**1**	**1**	**3**	**1**	
	0	0	0	0	0	0	0	0	0	0	0	0	2	**1**	3a
	0	0	0	0	0	0	0	0	0	0	0	0	2	2	
	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	3b
														…	
	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	3c
														…	
	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	3d
														…	
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	4
														…	

RLS Algorithm

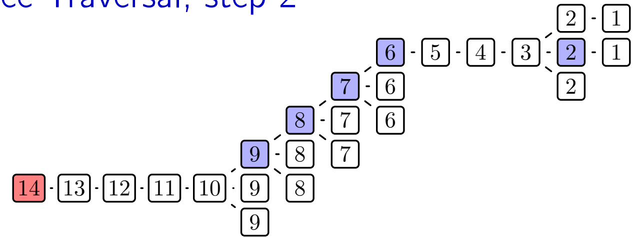
1. Set ESR $= (0, \ldots, 0)$
2. Run State Recursion using the current ESR
3. Save the number of equilibria in every stage game as *ne*(ESR)
4. Add 1 to the ESR in bases *ne*(ESR) to obtain new feasible ESR
5. Stopping rule: run out of digits
6. Return to step 2

RLS = Tree traversal

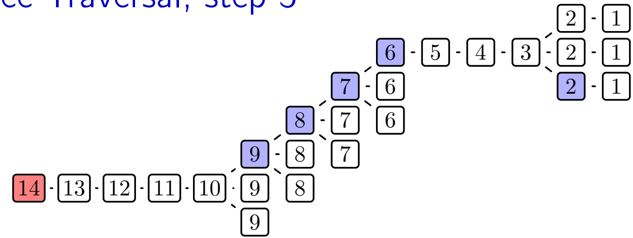
RLS Tree Traversal, step 1



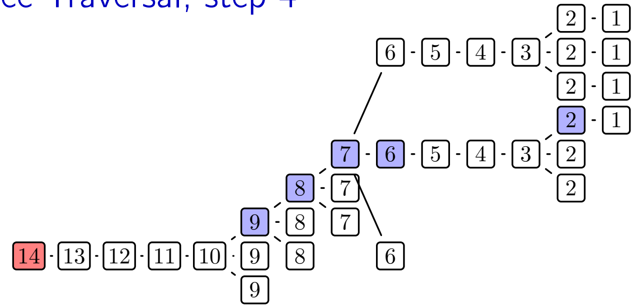
RLS Tree Traversal, step 2



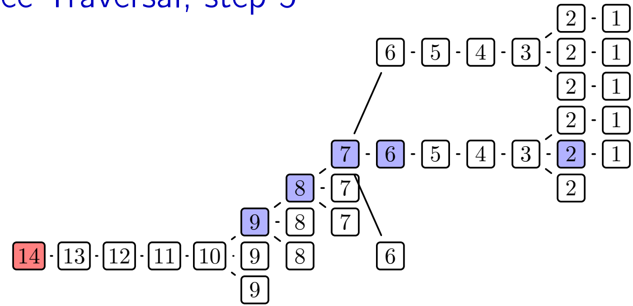
RLS Tree Traversal, step 3



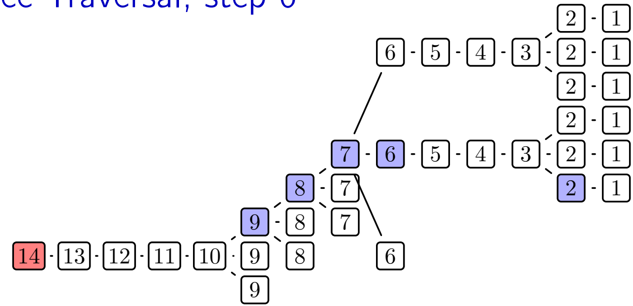
RLS Tree Traversal, step 4



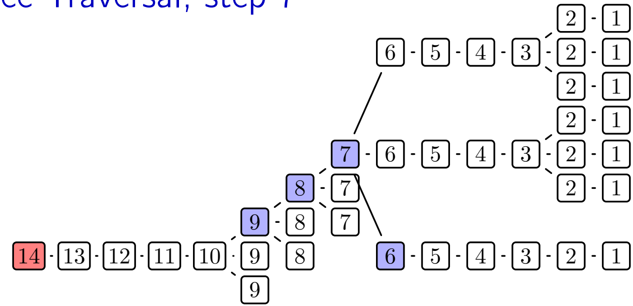
RLS Tree Traversal, step 5



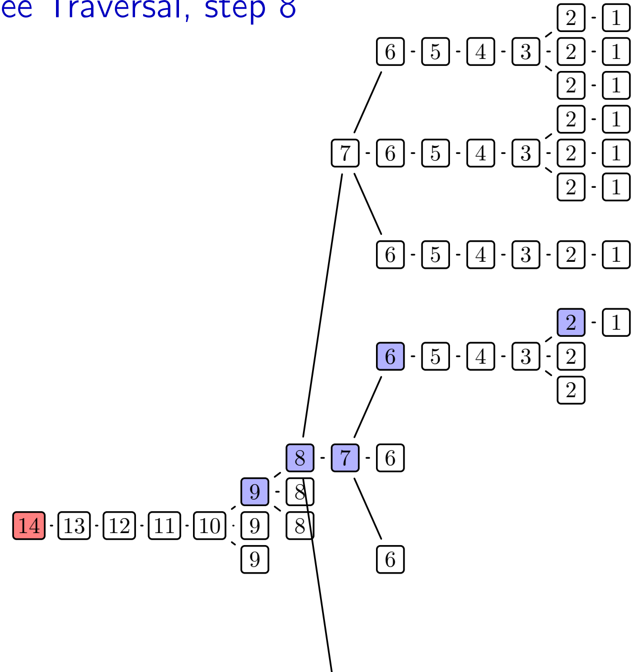
RLS Tree Traversal, step 6



RLS Tree Traversal, step 7



RLS Tree Traversal, step 8



Recursive Lexicographic Search (RLS) algorithm

Theorem (RLS theorem)

Assume there exists an algorithm that can find all MPE of every stage game of the DDG, and that the number of these equilibria is finite in every stage game.
Then the RLS algorithm finds all MPE of the DDG in a finite number of steps, which equals the total number of MPE.

Iskhakov, Rust and Schjerning, 2016, ReStud
"Recursive lexicographical search: Finding all markov perfect equilibria of finite state directional dynamic games."

ROAD MAP

1. Solving directional dynamic games (DDGs):
   - Simple example: Bertrand pricing and investment game
   - State recursion algorithm
   - Recursive lexicographical search (RLS) algorithm
2. Structural estimation of DDGs using Nested RLS
3. Refinements of NRLS: The need for speed
4. Monte Carlo: (Compare NRLS, two-step CCP, NPL, EPL, MPEC)

Nested Recursive Lexicographical Search (NRLS)

- Data from M independent markets from T periods
  $\mathrm{Z} = \{\bar{\mathrm{a}}^{mt}, \bar{\mathrm{x}}^{mt}\}_{m \in \mathcal{M}, t \in \mathcal{T}}$
  Usually assume only one equilibrium is played in the data.
- Denote $\big(\mathrm{P}^{\ell}(\theta), \mathrm{V}^{\ell}(\theta)\big) \in SOL(\Psi, \theta)$ the ℓ-the equilibrium

1. Outer loop
   Maximization of the likelihood function w.r.t. to structural parameters θ
   $$\theta^{ML} = \arg\max_{\theta} \mathcal{L}(Z, \theta)$$
2. Inner loop
   Maximization of the likelihood function w.r.t. equilibrium selection
   $$\mathcal{L}(Z, \theta) = \max_{(\mathrm{P}^{\ell}(\theta), \mathrm{V}^{\ell}(\theta)) \in SOL(\Psi, \theta)} \frac{1}{M} \sum_{i=1}^{N} \sum_{m=1}^{M} \sum_{t=1}^{T} \log P_i^{\ell}(\bar{a}_i{}^{mt} | \bar{\mathrm{x}}^{mt}; \theta)$$

Max of a function on a discrete set organized into RLS tree

Branch and bound (BnB) method

Land and Doig, 1960 *Econometrica*

- Old method for solving discrete programming problems

1. Form a tree of subdivisions of the set of admissible plans
2. Specify a bounding function representing the best attainable objective on a given subset (branch)
3. Dismiss the subsets of the plans where the bound is below the current best attained value of the objective

- **Branching**: RLS tree
- **Bounding**: The bound function is partial likelihood calculated on the subset of states

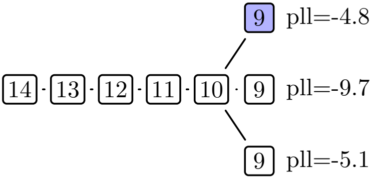
$$\mathcal{L}^{\text{Part}}(Z,\theta,\ell,\mathcal{S}) = \frac{1}{M}\sum_{i=1}^{N}\sum_{m=1}^{M}\sum_{t=1}^{T}\log P_i^{\ell}(\bar{a}_i{}^{mt}|\bar{\mathrm{x}}^{mt};\theta)$$
$$\text{s.t. } (\bar{\mathrm{x}}^{mt},\bar{a}_i{}^{mt}) \in \mathcal{S}$$

- Monotonically declines as more data is added
- Equals to the full log-likelihood at the leafs of RLS tree

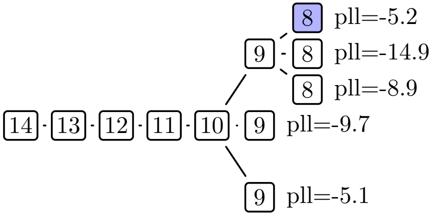
BnB on RLS tree, step 1

$\boxed{14} \cdot \boxed{13} \cdot \boxed{12} \cdot \boxed{11} \cdot \boxed{10}$ Partial loglikelihood = -3.2

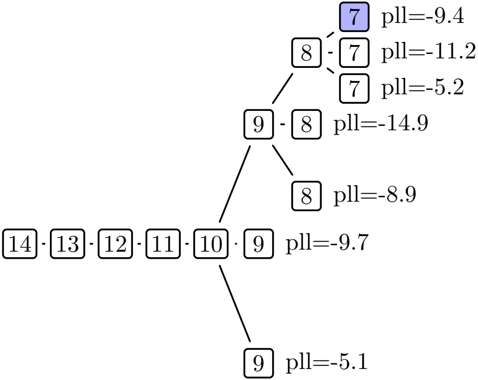
BnB on RLS tree, step 2



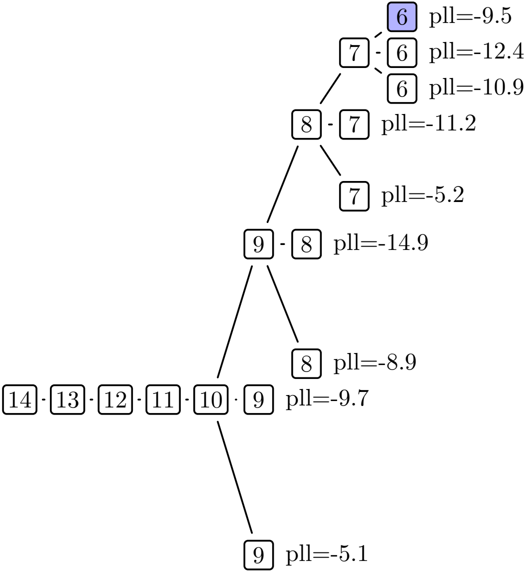
BnB on RLS tree, step 3



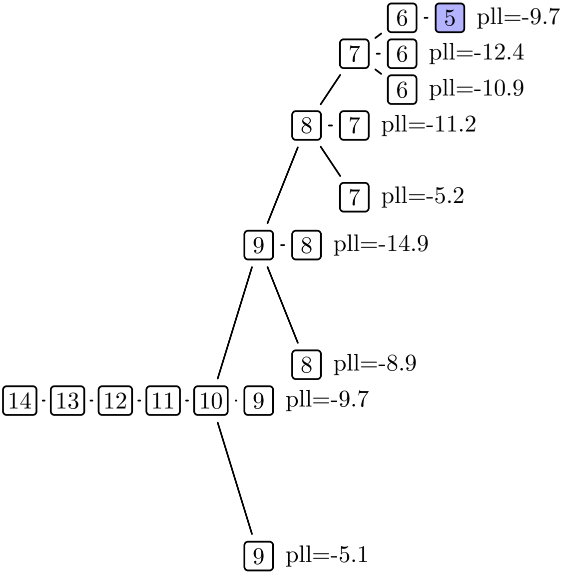
BnB on RLS tree, step 4



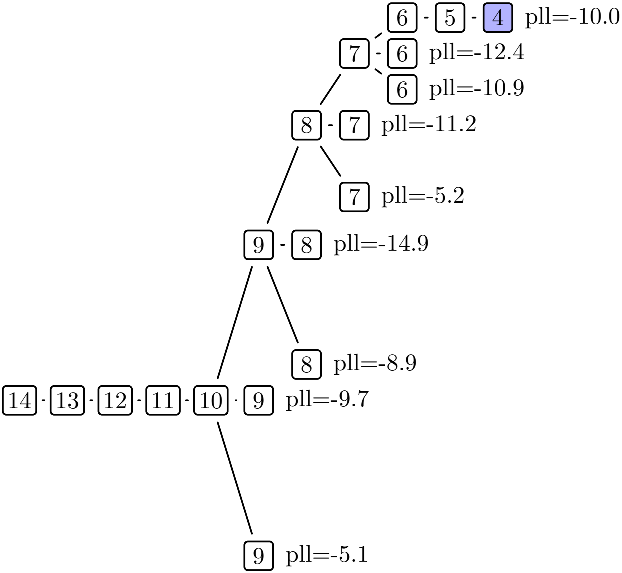
BnB on RLS tree, step 5



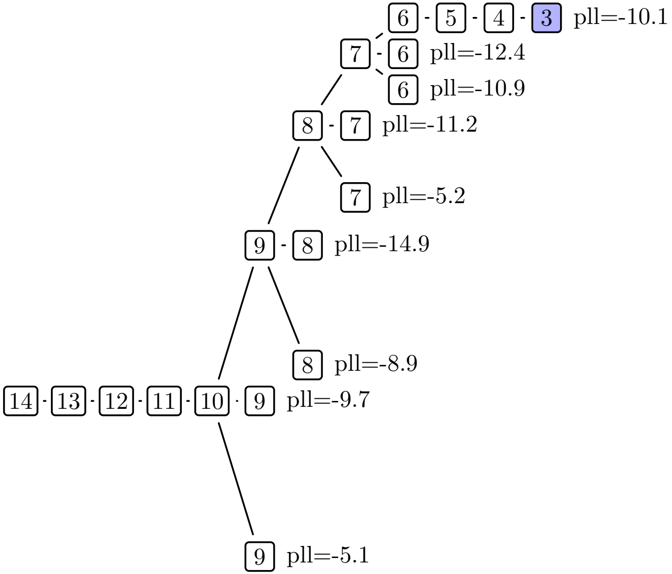
BnB on RLS tree, step 6



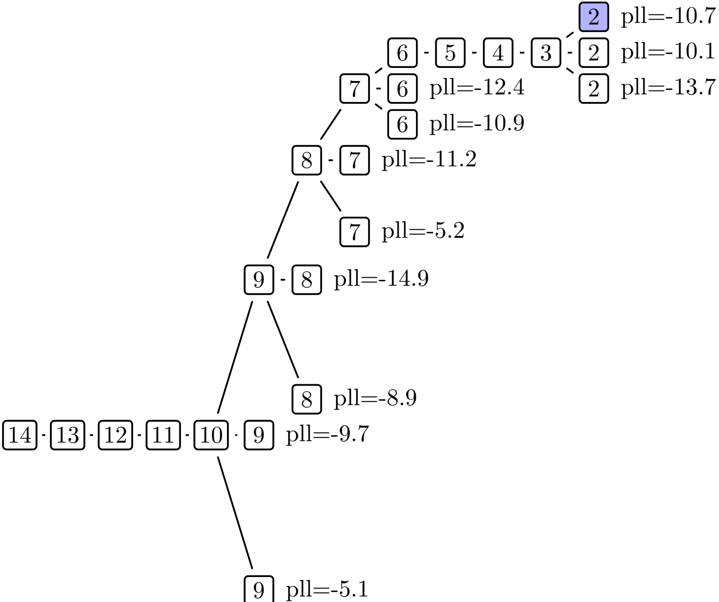
BnB on RLS tree, step 7



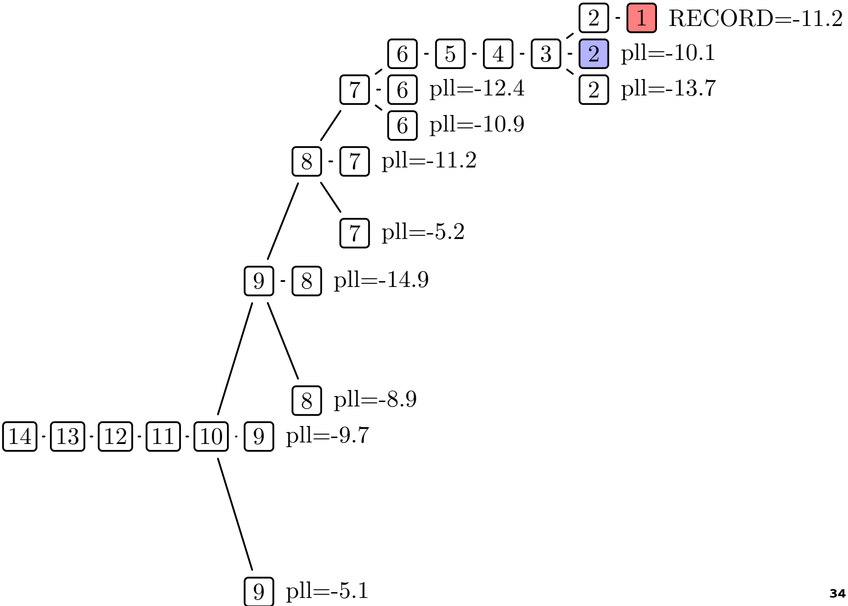
BnB on RLS tree, step 8



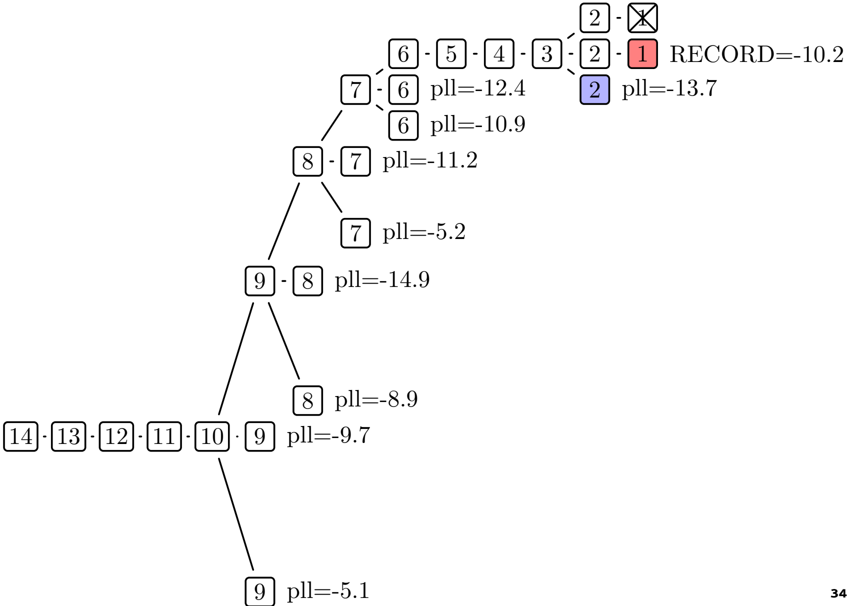
BnB on RLS tree, step 9



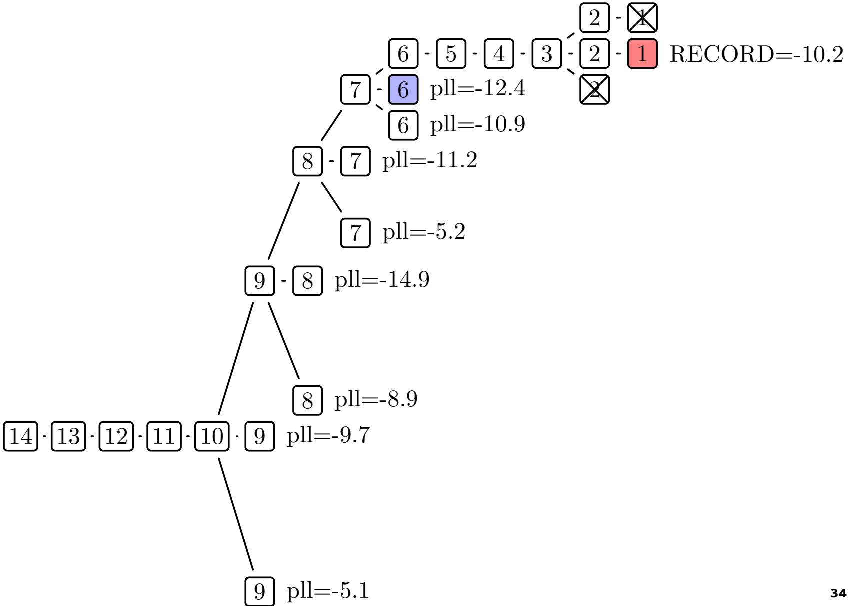
BnB on RLS tree, step 10



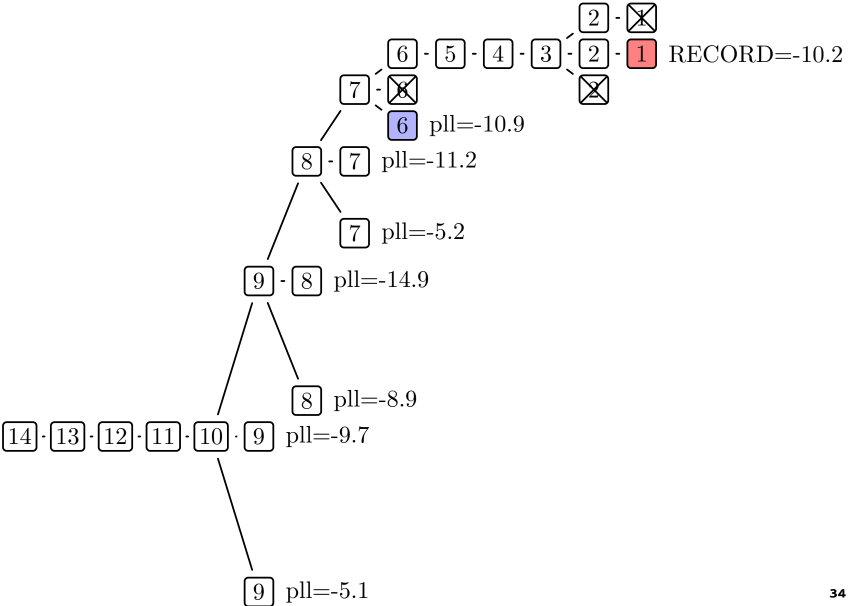
BnB on RLS tree, step 11



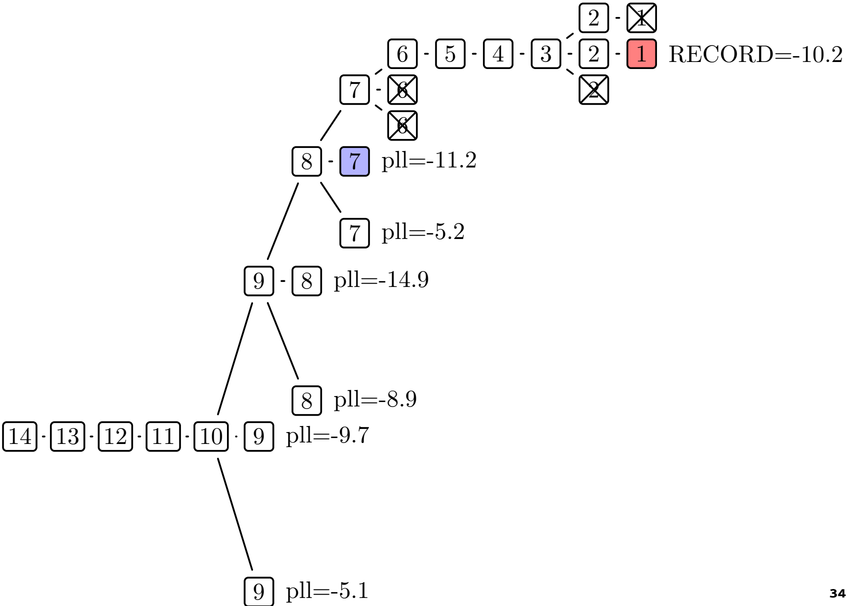
BnB on RLS tree, step 12



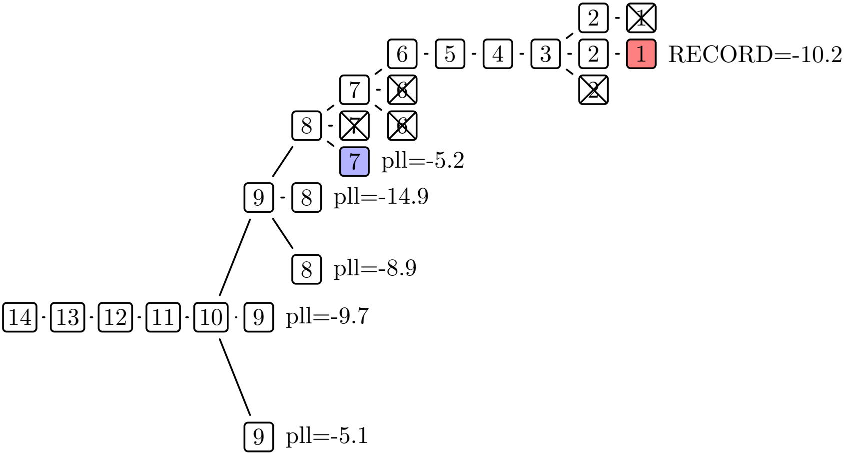
BnB on RLS tree, step 13



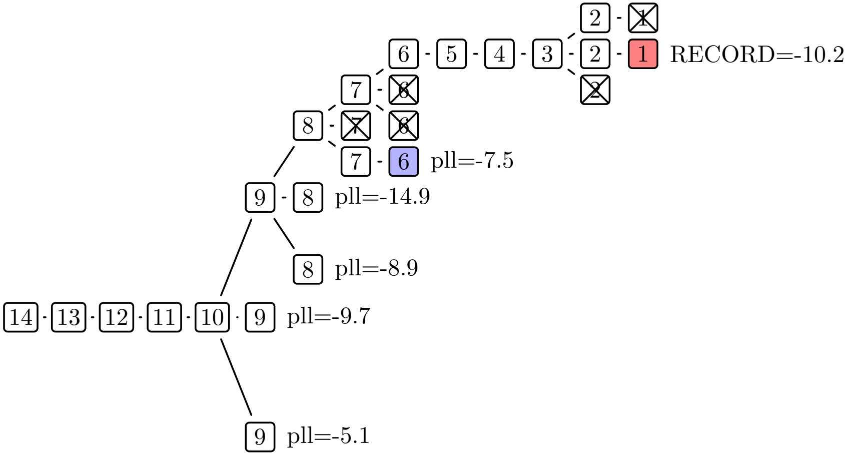
BnB on RLS tree, step 14



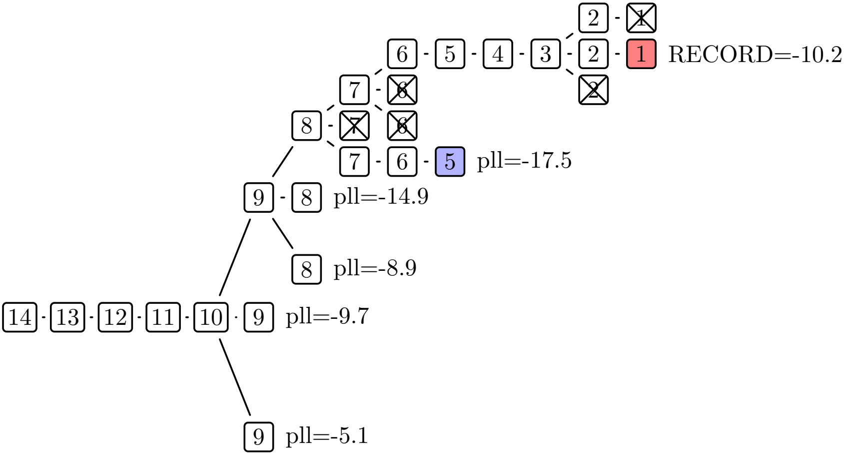
BnB on RLS tree, step 15



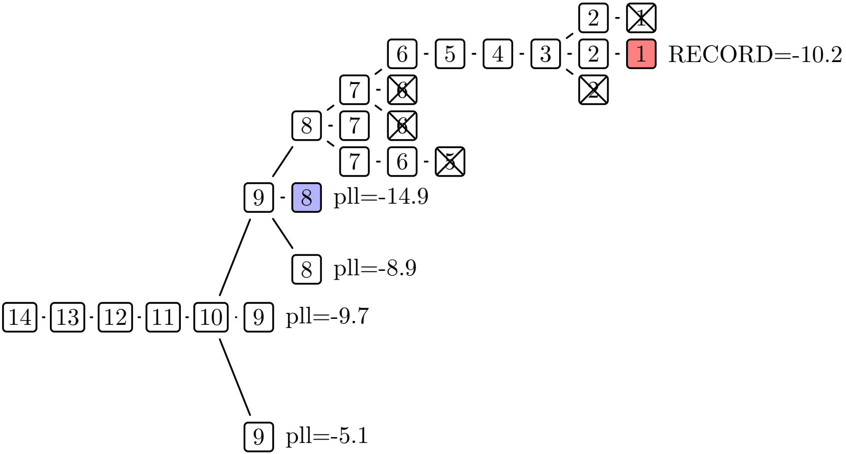
BnB on RLS tree, step 16



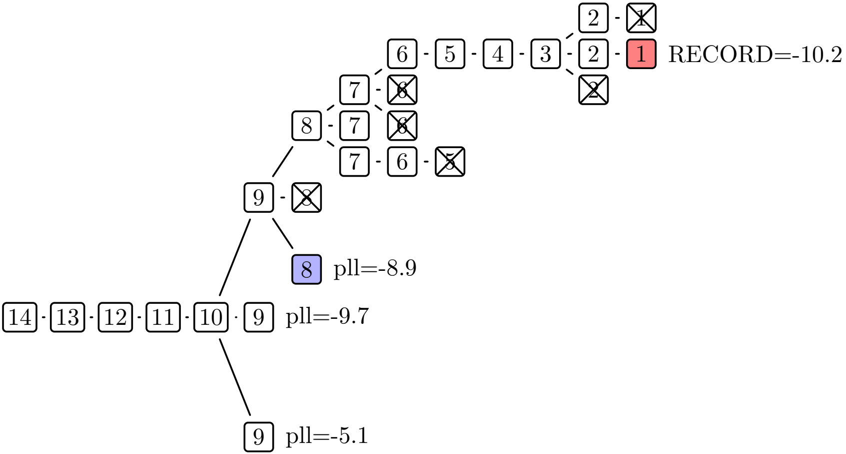
BnB on RLS tree, step 17



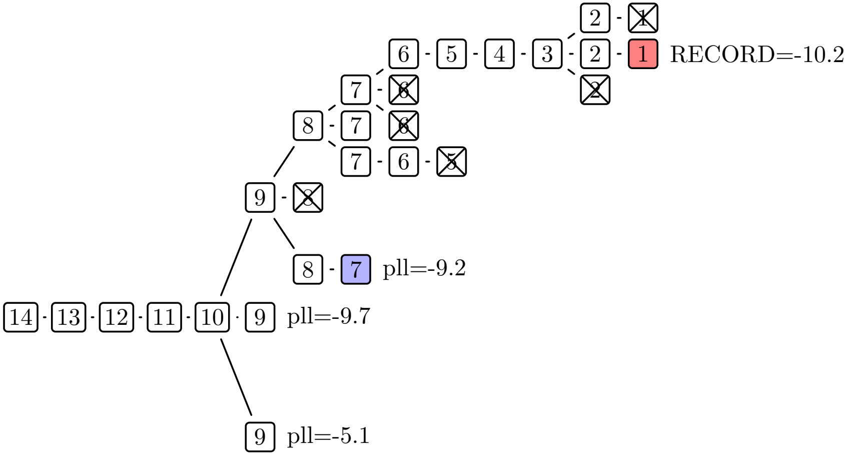
BnB on RLS tree, step 18



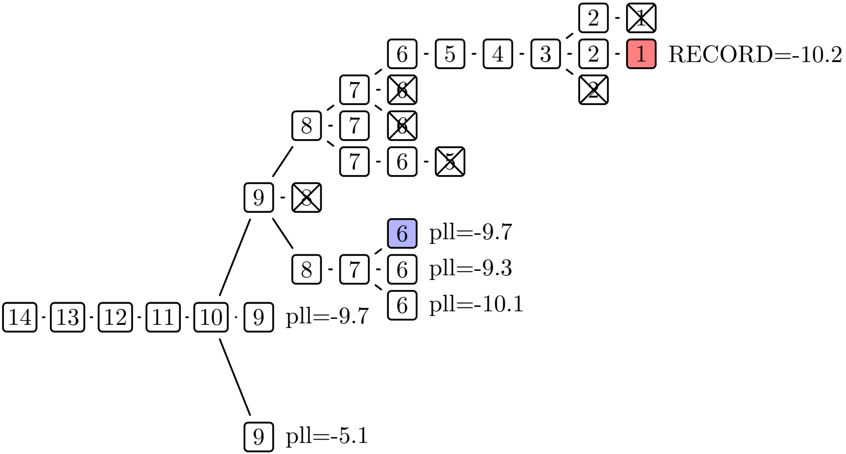
BnB on RLS tree, step 19



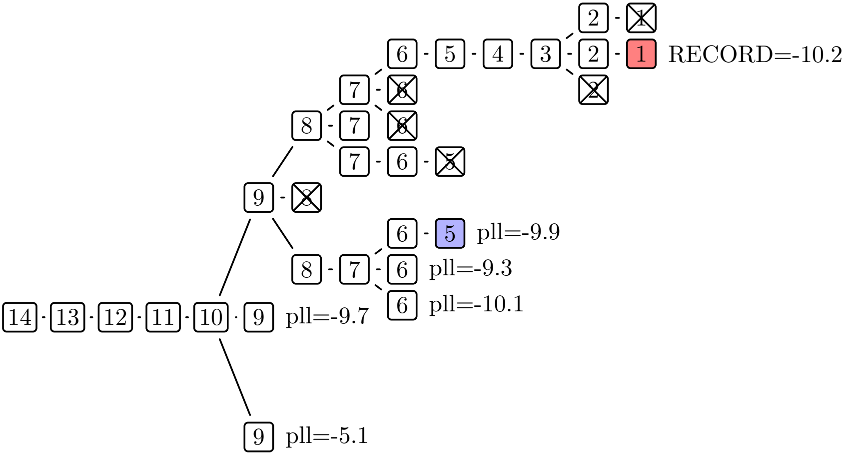
BnB on RLS tree, step 20



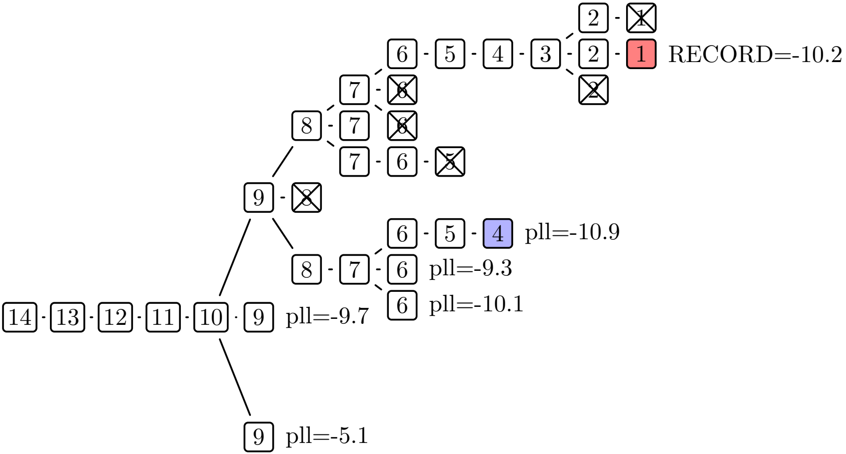
BnB on RLS tree, step 21



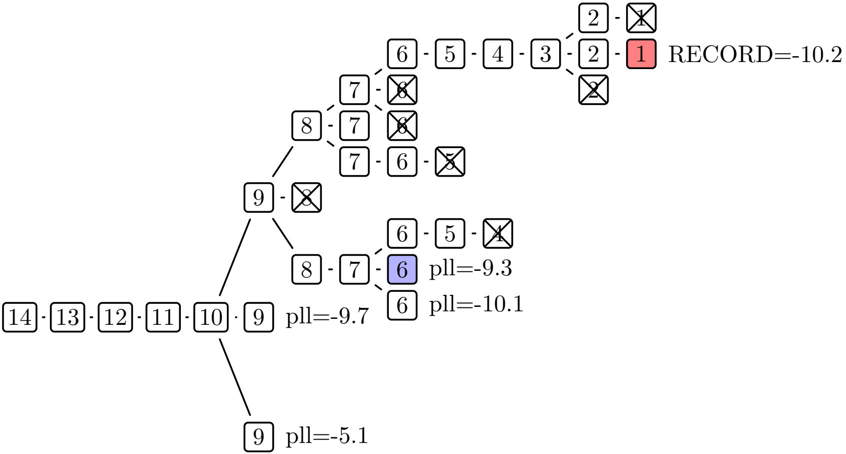
BnB on RLS tree, step 22



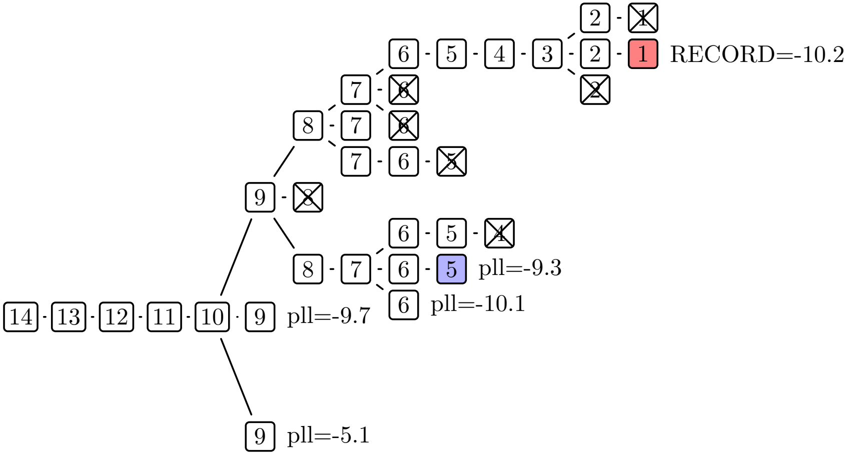
BnB on RLS tree, step 23



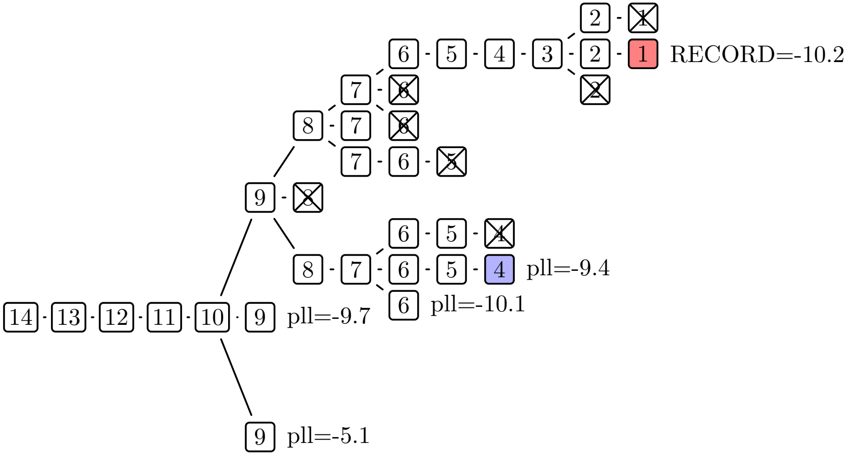
BnB on RLS tree, step 24



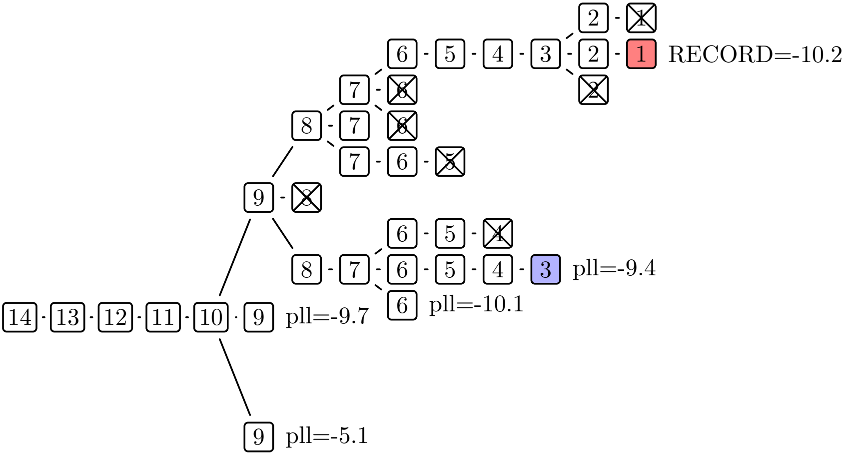
BnB on RLS tree, step 25



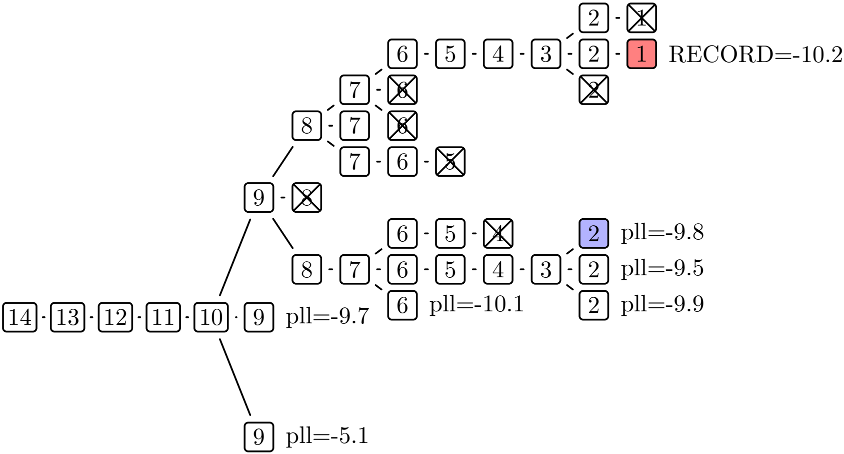
BnB on RLS tree, step 26



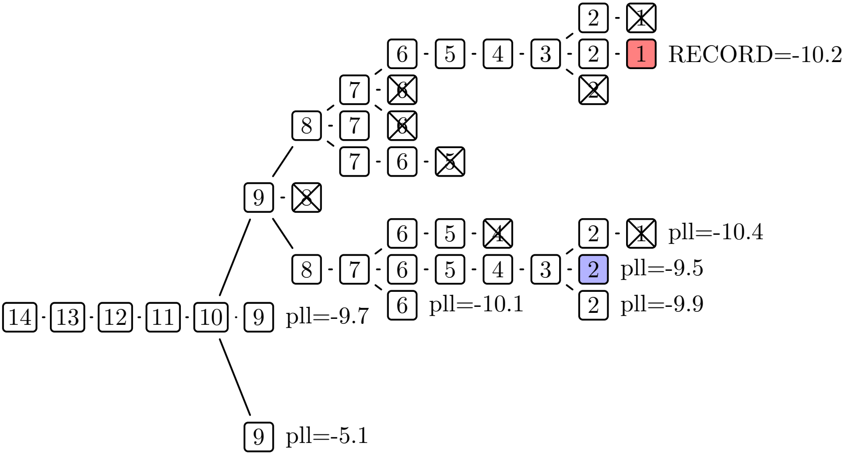
BnB on RLS tree, step 27



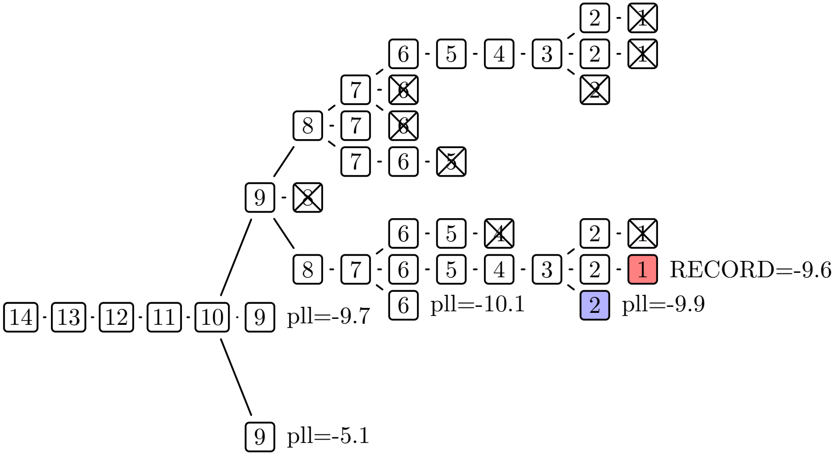
BnB on RLS tree, step 28



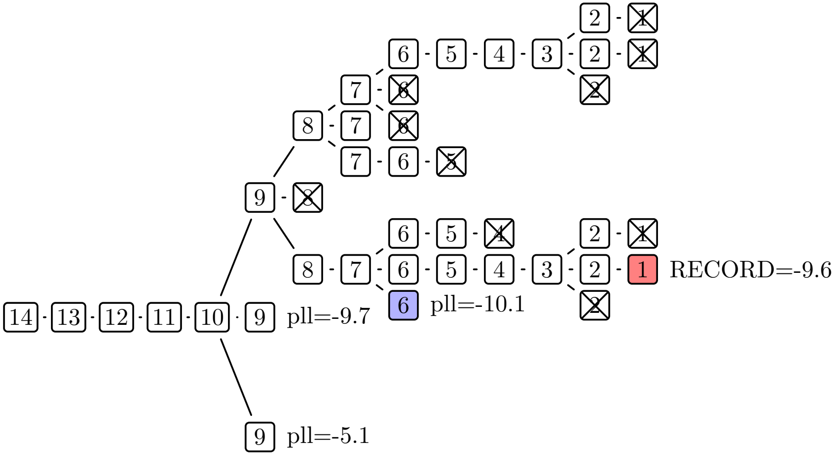
BnB on RLS tree, step 29



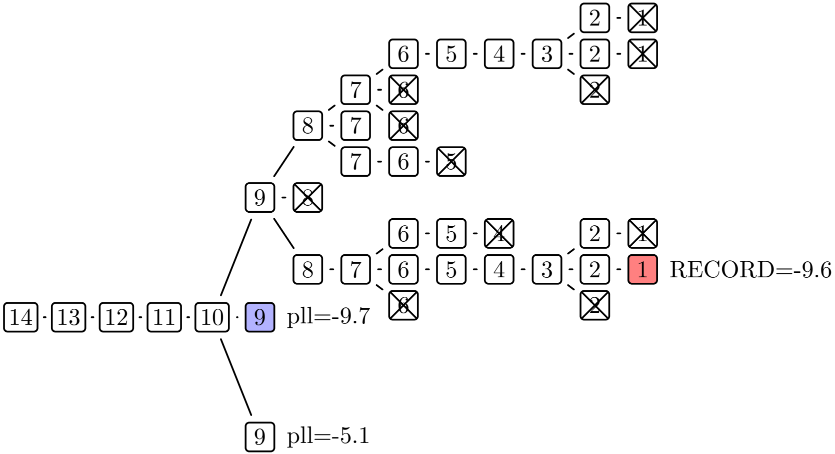
BnB on RLS tree, step 30



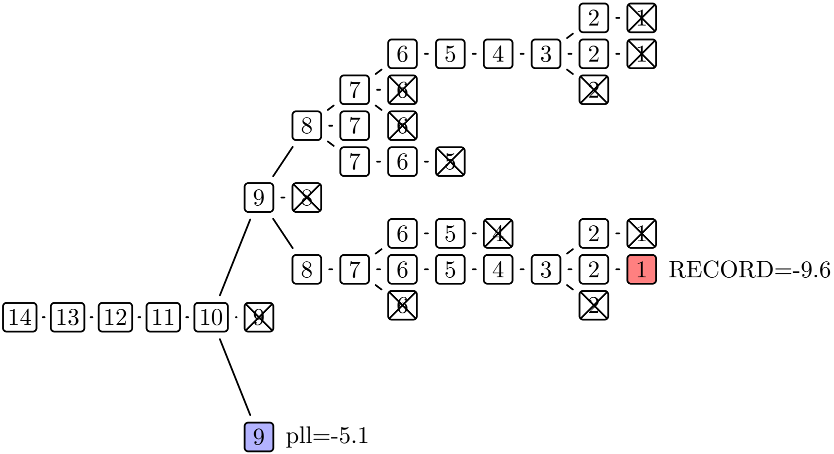
BnB on RLS tree, step 31



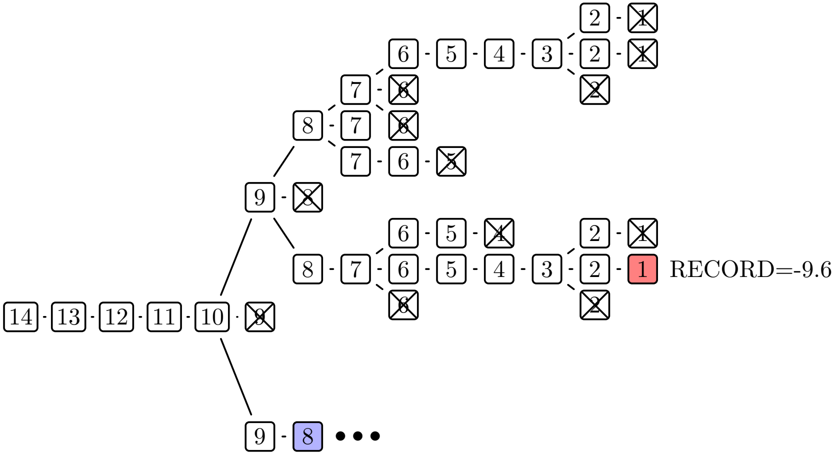
BnB on RLS tree, step 32



BnB on RLS tree, step 33



BnB on RLS tree, step 34



ROAD MAP

1. Solving directional dynamic games (DDGs):
   - Simple example: Bertrand pricing and investment game
   - State recursion algorithm
   - Recursive lexicographical search (RLS) algorithm
2. Structural estimation of DDGs using Nested RLS
3. Refinements of NRLS: The need for speed
4. Monte Carlo: (Compare NRLS, two-step CCP, NPL, EPL, MPEC)

Partial Likelihood on a subset of the state space

- Likelihood contribution from state point i, and MPE ω

$$L_i(\theta,\omega) = \sum_{j=1}^{2} \left[n_j^I(i) \log P_j^I(i,\theta,\omega) + n_j^N(i) \log P_j^N(i,\theta,\omega) \right],$$

- $k = 1$ is the initial point (the apex in the leapfrogging game)
- $k = K$ is to the terminal point (an absorbing stage, e.g. 0,0,0)

- Partial Likelihood at node k:

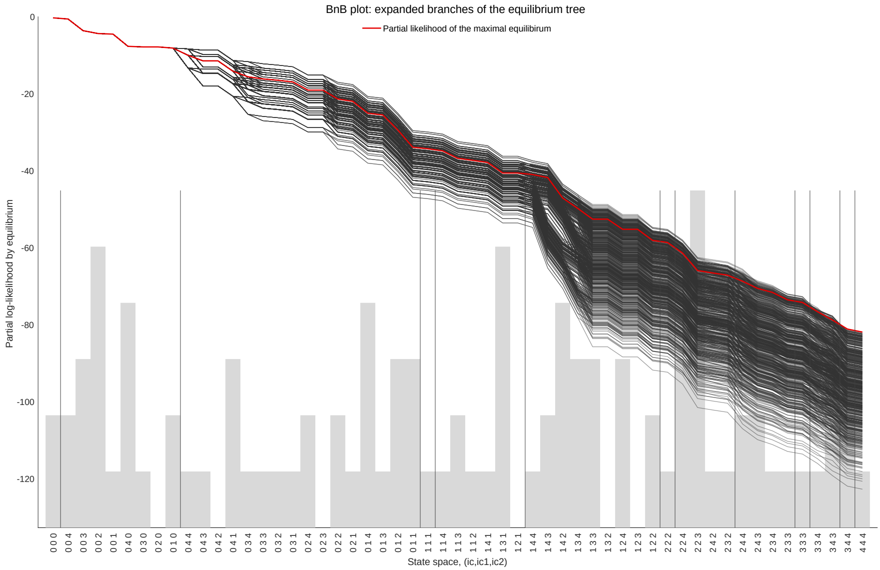
$$L(k,\theta,\omega) = \sum_{i=k}^{K} L_i(\theta,\omega)$$

  - The partial likelihood accumulates data likelihood from the terminal point K to state k
  - Multiple equilibria from a node share partial likelihood up to the previous point in the RLS tree.

- Likelihood Function: $L(\theta,\omega) = \sum_{i=1}^{K} L_i(\theta,\omega) = L(1,\theta,\omega)$

BnB and partial likelihood

$k = K = 14$ (terminal state) on the left, $k = 1$ (initial state) on the right



Non-parametric likelihood Line

- Partial non-parametric Log-Likelihood:

$$L_i^e = \sum_{j=1}^{2} \left[n_j^I(i) \log \frac{n_j^I(i)}{n_j^I(i) + n_j^N(i)} + n_j^N(i) \log \frac{n_j^N(i)}{n_j^I(i) + n_j^N(i)} \right]$$

- Non-parametric Likelihood Function:

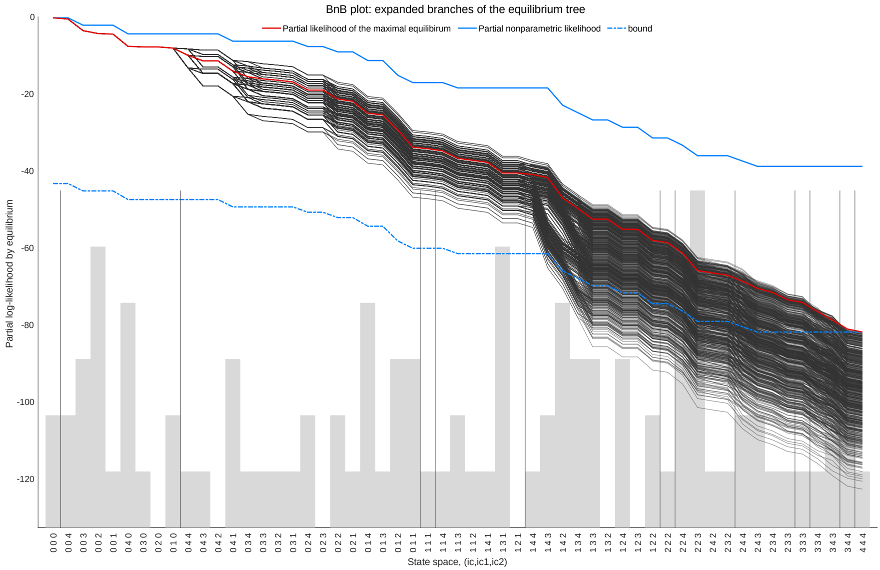
$$L^e = \sum_{i=1}^{K} L_i^e$$

- Remaining Non-parametric Likelihood at Node k:

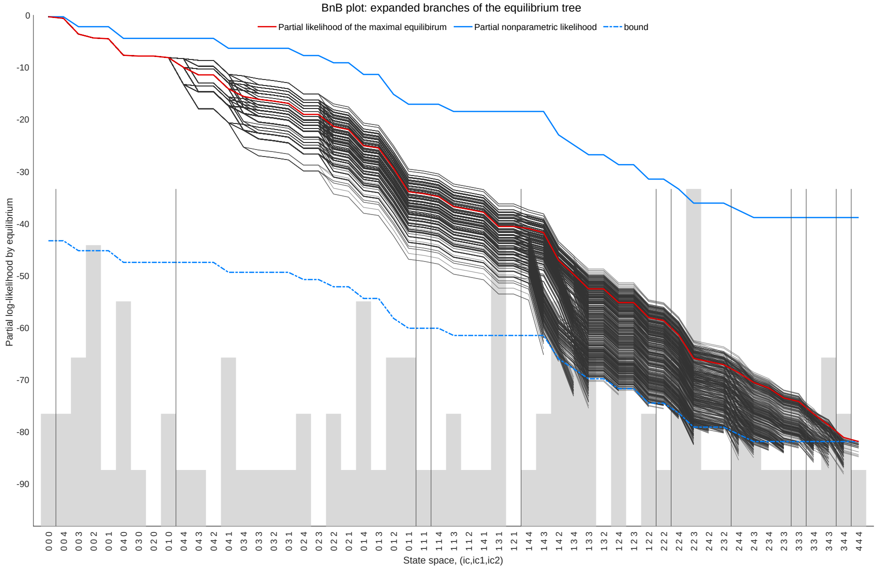
$$RL^e(k) = \sum_{i=1}^{k-1} L_i^e$$

  - Non-parametric likelihood is computed independently of structural parameters and equilibrium selection.
  - The remaining Non-parametric likelihood at any node k is the sum of empirical likelihoods for unaccounted data up to $k-1$.

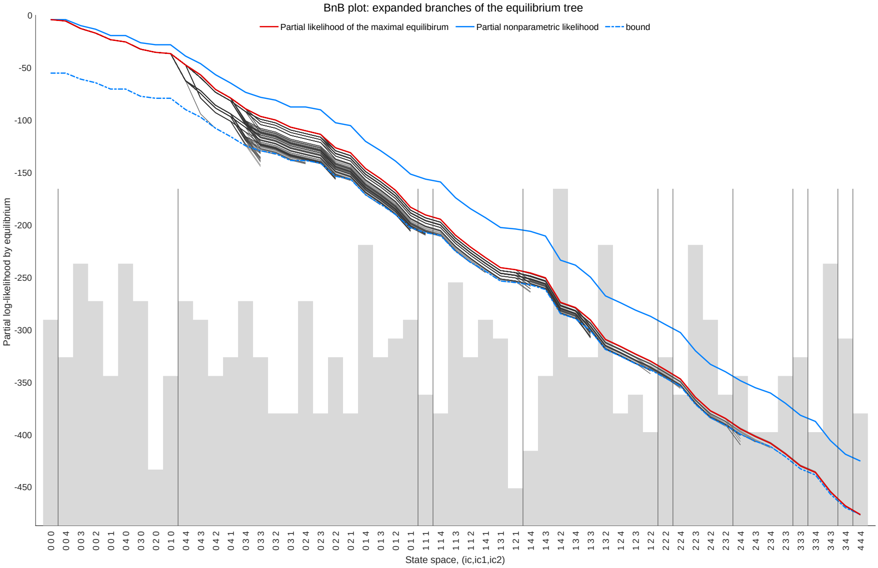
Non-parameteric remaining likelihood



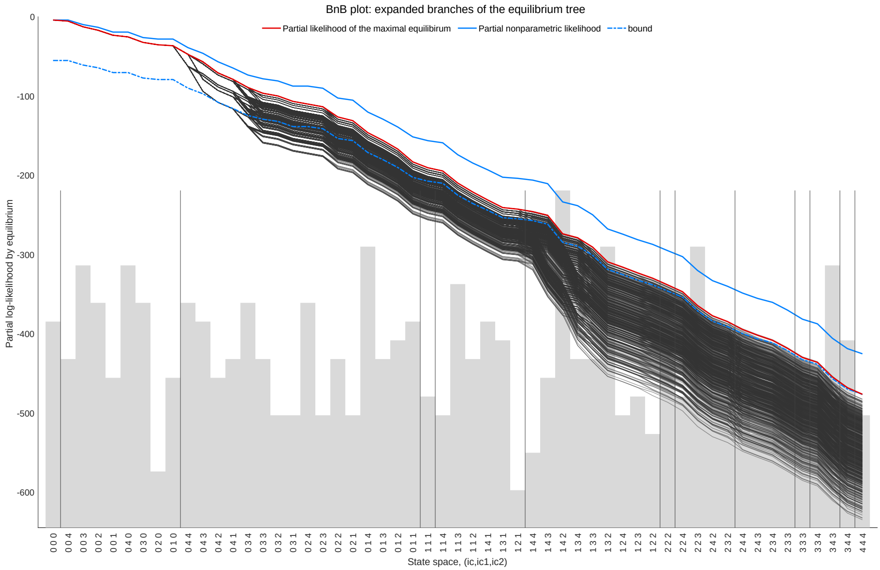
BnB with non-parameteric likelihood bound



BnB with non-parameteric likelihood bound, larger sample



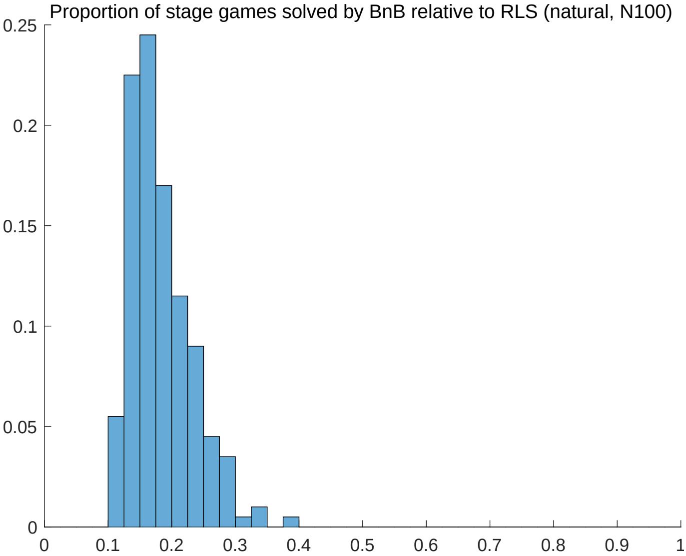
Full enumeration RLS in larger sample



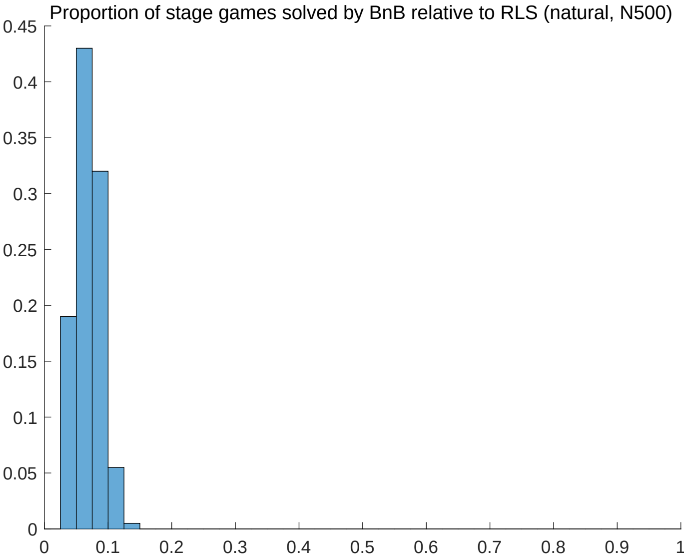
Remarks on numerical performance

- BnB refinement: BnB augmented with non-parameteric likelihood bound gives sharper Bounding Rules $\rightarrow$ less computation
- More data:
  - Non-parametric log-likelihood converge to the likelihood line.
  - The width of the band between the blue lines in the plots decreases with increasing sample size
    $\rightarrow$ Sharper Bounding Rules
    $\rightarrow$ Less computation

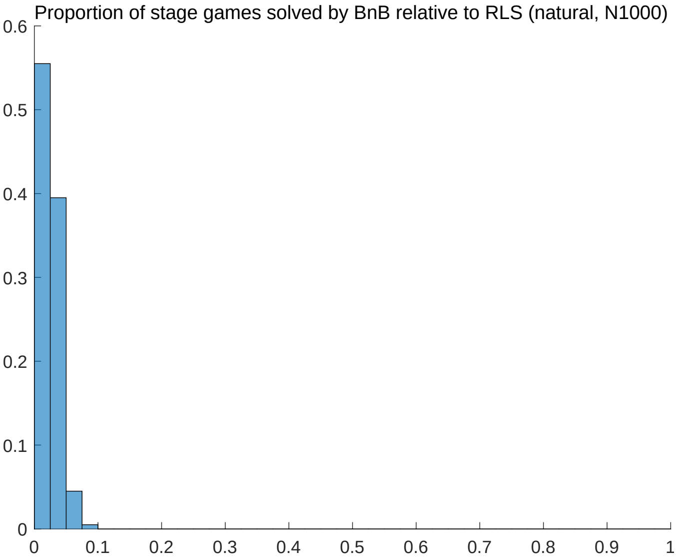
Fraction of stage games solved, $N = 100$



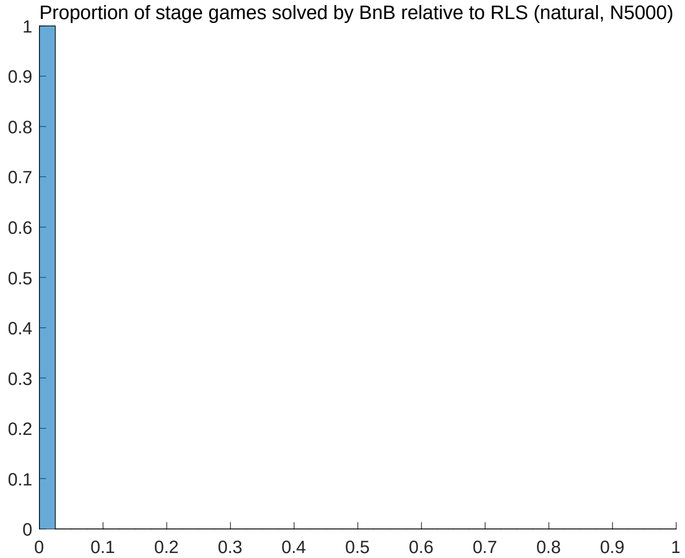
Fraction of stage games solved, $N = 500$



Fraction of stage games solved, $N = 1000$

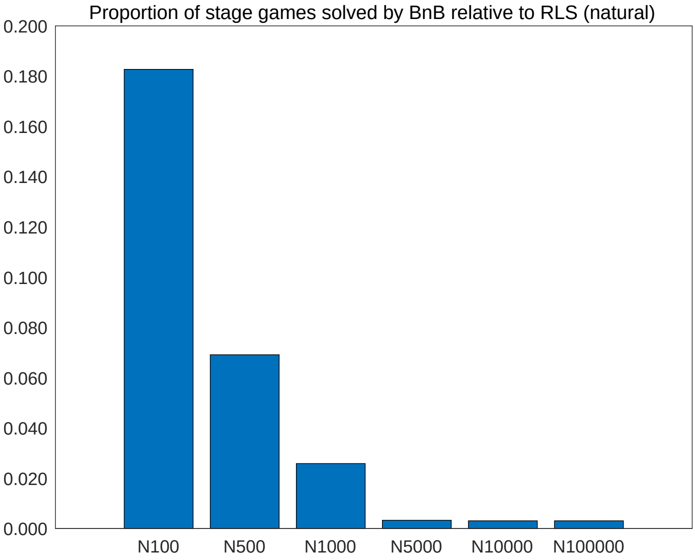


Fraction of stage games solved, $N = 5000$



Reduction in the number of stage games to solve

As sample size increases, computational burden decreases sharply



ROAD MAP

1. Solving directional dynamic games (DDGs):
   - Simple example: Bertrand pricing and investment game
   - State recursion algorithm
   - Recursive lexicographical search (RLS) algorithm
2. Structural estimation of DDGs using Nested RLS
3. Refinements of NRLS: The need for speed
4. Monte Carlo: (Compare NRLS, two-step CCP, NPL, EPL, MPEC)

Monte Carlo simulations

A

Single equilibrium in the model
Single equilibrium in the data

B

Multiple equilibria in the model
Single equilibrium in the data

C

Multiple equilibria in the model
Multiple equilibria in the data

1. Two-step CCP estimator
2. Nested pseudo-likelihood vs. NRLS estimator
3. MPEC

Implementation details

- Two-step estimator and NPL
  - Matlab unconstraint optimizer (numerical derivatives)
  - CCPs from frequency estimators
  - For NPL max 30 iterations
- MPEC
  - Matlab constraint optimizer (interior-point algorithm)
  - MPEC-VP: Constraints on both values and choice probabilities (as in Egesdal, Lai and Su, 2015)
  - MPEC-P: Constraints in terms of choice probabilities + Hotz-Miller inversion
  - Starting values from two-step estimator
- Estimated parameters $\theta = (k_1, k_2)$
- Sample size: 1000 markets in 5 time periods
- Initial state drawn uniformly over the state space

Monte Carlo A, run 1: no multiplicity

Maximum number of equilibria in the model: 1

Number of equilibria in the data: 1

	PML2step	NPL	MPEC-VP	MPEC-P	NRLS
k1=3.5	3.51893	3.51022	3.50380	3.50380	3.50380
Bias	0.01893	0.01022	0.00380	0.00380	0.00380
MCSD	0.12087	0.12635	0.11573	0.11573	0.11573
k2=0.5	0.50860	0.50658	0.50452	0.50452	0.50452
Bias	0.00860	0.00658	0.00452	0.00452	0.00452
MCSD	0.06460	0.06247	0.05939	0.05939	0.05939
log-likelihood	-1958.176	-1953.406	-1953.327	-1953.327	-1953.327
$\|\|\Psi^{\mathbf{P}}(P) - P\|\|$	0.25285	0.00001	0.00000	0.00000	0.00000
$\|\|\Psi^{\mathbf{V}}(v) - v\|\|$	0.50038	0.00001	0.00000	0.00000	0.00000
Converged,%	100	100	100	100	100
K-L divergence	0.131139	0.005020	0.006770	0.006770	0.006770

- All MLE estimators identical to the last digit
- NPL estimator is approaching MLE

Monte Carlo A, run 2: no multiplicity at true parameter

Maximum number of equilibria in the model: 3

Number of equilibria at true parameter value: 1

Number of equilibria in the data: 1

	PML2step	NPL	MPEC-VP	MPEC-P	NRLS
k1=3.5	3.50467	3.51307	3.49485	3.49318	3.49318
Bias	0.00467	0.01307	-0.00515	-0.00682	-0.00682
MCSD	0.11252	0.00000	0.10193	0.10177	0.10177
k2=0.5	0.50035	0.47394	0.50265	0.50157	0.50157
Bias	0.00035	-0.02606	0.00265	0.00157	0.00157
MCSD	0.05009	0.00000	0.04154	0.04205	0.04205
log-likelihood	-4106.771	-3940.158	-4091.873	-4093.040	-4093.04
$\|\|\Psi^{\mathbf{P}}(P) - P\|\|$	0.41453	0.00001	0.00000	0.00000	0.00000
$\|\|\Psi^{\mathbf{V}}(v) - v\|\|$	1.90182	0.00005	0.00000	0.00000	0.00000
Converged,%	100	1	98	100	100
K-L divergence	0.188551	0.004546	0.002921	0.002921	0.002920

- NPL estimator fails to converge
- MPEC is not affected by “nearby” equilibria with good starting values (PML2step)

Monte Carlo B, run 1: moderate multiplicity

Number of equilibria in the model (at true parameter): 3
Number of equilibria in the data: 1

	PML2step	NPL	MPEC-VP	MPEC-P	NRLS
k1=3.5	3.50081	-	3.72713	3.94941	3.49624
Bias	0.00081	-	0.22713	0.44941	-0.00376
MCSD	0.12050	-	0.85934	1.16633	0.09537
k2=0.5	0.49478	-	0.56166	0.62361	0.49381
Bias	-0.00522	-	0.06166	0.12361	-0.00619
MCSD	0.04317	-	0.25552	0.32488	0.03510
log-likelihood	-4070.035	-	-4080.989	-4121.102	-4049.647
$\|\|\Psi^{\mathbf{P}}(P) - P\|\|$	0.50375	-	0.00000	0.00000	0.00000
$\|\|\Psi^{\mathbf{V}}(v) - v\|\|$	2.83611	-	0.00000	0.00000	0.00000
Converged,%	100	0	100	100	100
K-L divergence	0.304411	-	0.018636	2.302525	0.006314

- NPL estimator fails to converge
- MPEC fails to identify the equilibrium that generated the data (converges to a different MPE) as seen from MCSD and K-L divergence
- MPEC get stuck in local minima (constraints are satisfied, but likelihood is low)

Monte Carlo B, run 2: higher multiplicity

Number of equilibria in the model (at true parameter): 81

Number of equilibria in the data: 1

	PML2step	NPL	MPEC-VP	MPEC-P	NRLS
k1=3.5	3.51468	-	3.48740	3.49007	3.47786
Bias	0.01468	-	-0.01260	-0.00993	-0.02214
MCSD	0.04844	-	0.02802	0.02929	0.02731
k2=0.5	0.53780	-	0.49197	0.48944	0.49252
Bias	0.03780	-	-0.00803	-0.01056	-0.00748
MCSD	0.03894	-	0.00850	0.01033	0.00404
log-likelihood	-4038.78471	-	-4007.45663	-4010.18139	-3996.45223
$\|\|\Psi^{\mathbf{P}}(P) - P\|\|$	0.68907	-	0.00000	0.00000	0.00000
$\|\|\Psi^{\mathbf{V}}(v) - v\|\|$	5.44052	-	0.00000	0.00000	0.00000
Converged,%	100	0	100	100	100
K-L divergence	0.453917	-	0.278263	0.356678	0.000750

- NPL estimator fails to converge
- MPEC fails to identify the DGP equilibrium (converges to a different MPE)
- With good starting values, does not suffer more with higher multiplicity

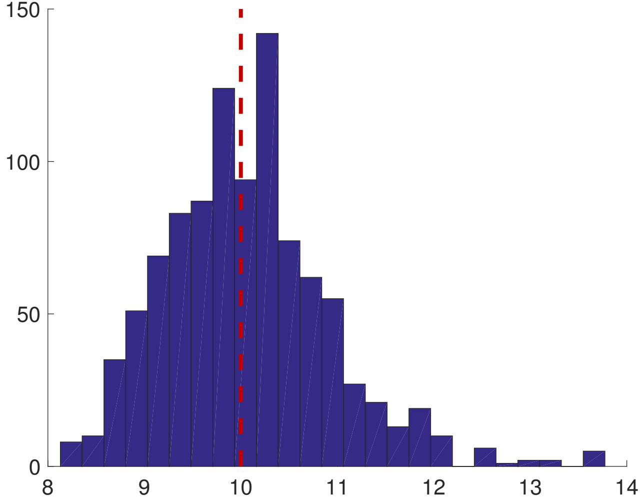
NRLS Monte Carlo setup (C)

- $n = 3$ points on the grid on the grid of costs
- 14 points in state space of the model
- 109 MPE in total

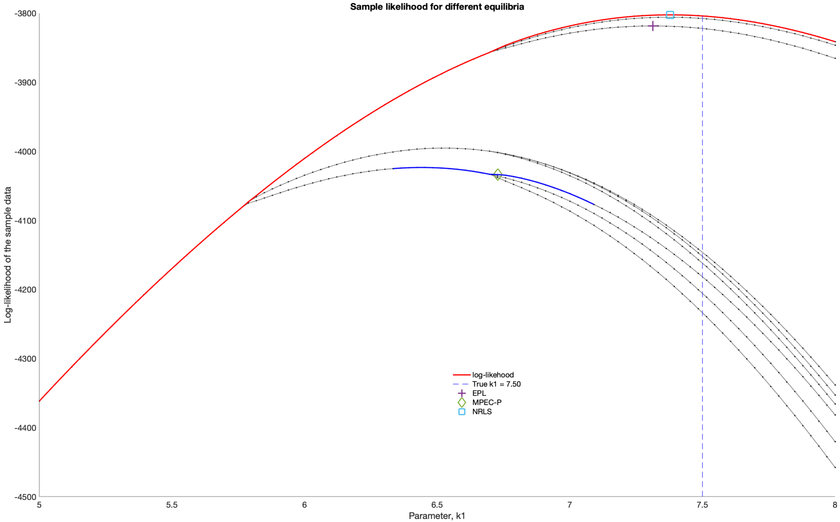
- 1000 random samples from 3 different equilibria (3 markets)
- 100 observations per market/equilibrium
- Uniform distribution over state space $\leftrightarrow$ "ideal" data

- Estimating one parameter in cost function

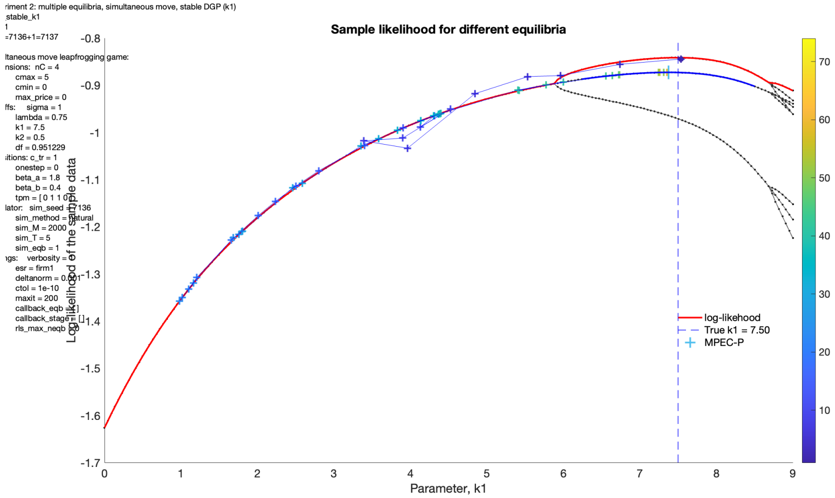
Distribution of estimated k_1 parameter



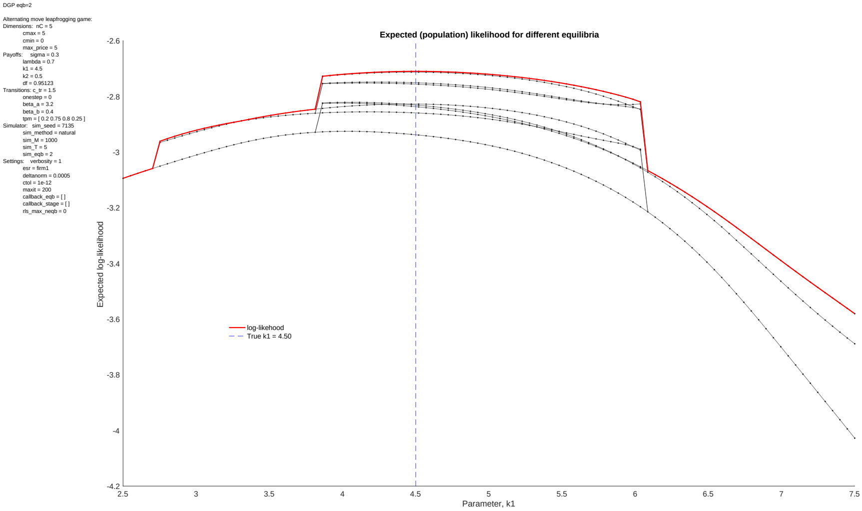
Failure of existing estimators, local maxima



Convergence to local maximum for MPEC-P



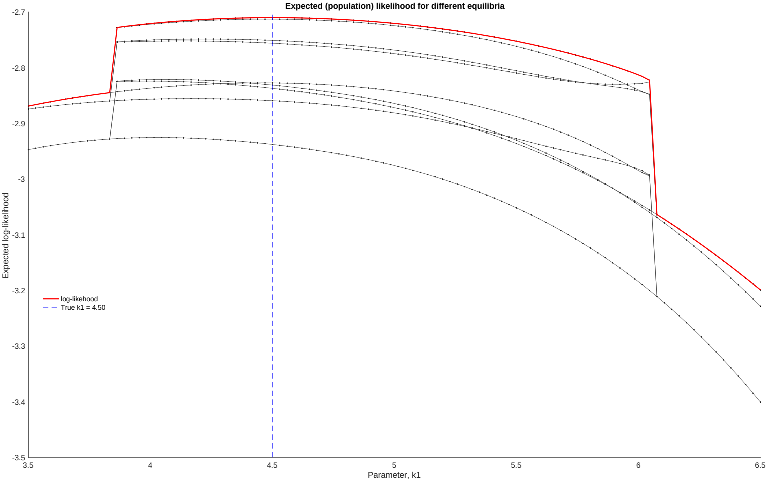
Discontinuous likelihood function



Discontinuous likelihood function

DGP eqb=2

Alternating move leapfrogging game:
Dimensions: nC = 5
cmax = 5
cmin = 0
max_price = 5
Payoffs: sigma = 0.3
lambda = 0.7
k1 = 4.5
k2 = 0.5
df = 0.95123
Transitions: c_tr = 1.5
onestep = 0
beta_a = 3.2
beta_b = 0.4
tpm = [0.2 0.75 0.8 0.25]
Simulator: sim_seed = 7135
sim_method = natural
sim_M = 1000
sim_T = 5
sim_eqb = 2
Settings: verbosity = 1
esr = firm1
deltanorm = 0.0005
ctol = 1e-12
maxit = 200
callback_eqb = [ ]
callback_stage = [ ]
rls_max_neqb = 0



NRLS estimator for directional dynamic games

Complicated computational task involving maximization over the large finite set of all MPE equilibria → branch-and-bound algorithm with refined bounding rule.

NRLS nested structure:

1. Each stage game → non-linear solver, specific to the model
2. Combining stage game solutions to full game MPEs → State Recursion algorithm
3. Solving for all MPE equilibria → Recursive Lexicographic Search
4. Structural estimation → high-dimensional optimization algorithm

Performance of NRLS

- Implementation of statistically efficient estimator (MLE)
- Using BnB NRLS avoids full enumeration at no cost.
- BnB augmented with non-parameteric likelihood bound gives sharper Bounding Rules → less computation
- Computationally trackable, better performance with more data
- Fully robust to multiplicity of equilibria in both data and the model