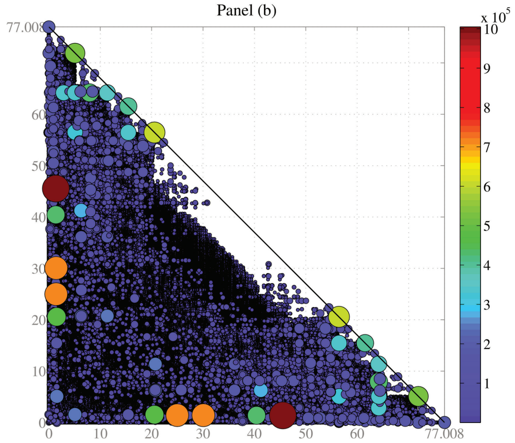
Structural Estimation of Directional Dynamic Games With Multiple Equilibria

Fedor Iskhakov, Australian National University
Dennis Kristensen, University College London
John Rust, Georgetown University
Bertel Schjerning, University of Copenhagen

Monash University
16 April 2025

Simultaneous move leapfrogging: 164,295,079 equilibria



Iskhakov, Rust and Schjerning (2016) *Review of Economic Studies*
"Recursive Lexicographical Search: Finding All Markov Perfect Equilibria of Finite State Directional Dynamic Games"

Estimation of stochastic dynamic games

1. Several decision makers (*players*)
2. Maximize discounted expected lifetime utility
3. Anticipate consequences of their current actions
4. Anticipate actions by other players in current and future periods (*strategic interaction*)
5. Operate in a stochastic environment (*state of the game*) whose evolution depend on the collective actions of the players

- Estimate structural parameters of these models
- Data on M independent markets over T periods
- Multiplicity of equilibria

Markov Perfect Equilibria

- Discrete-time infinite-horizon dynamic stochastic games with discrete states and actions
- MPE is a pair of strategy profiles and value functions such that

$$V = \Psi^V(V, P, \theta) \qquad \text{(Bellman equations)}$$
$$P = \Psi^P(V, P, \theta) \qquad \text{(CCPs = mutual best responces)}$$

- $\Psi = \left(\Psi^V, \Psi^P\right)$ gives the structure of the model
- Denote the set of all equilibria in the model as

$$\mathcal{E}(\Psi, \theta) = \left\{ (P, V) \middle| \begin{array}{c} V = \Psi^V(V, P, \theta) \\ P = \Psi^P(V, P, \theta) \end{array} \right\}$$

- Vision: Solve for all MPE equilibria for any θ

Maximum Likelihood Estimation

- Data from M independent markets from T periods

$$Z = \left\{ \bar{a}^{mt}, \bar{x}^{mt} \right\}_{m \in \mathcal{M}, t \in \mathcal{T}}$$

- Assume that only one equilibrium is played in the data (we relax this assumption later $\rightarrow$ grouped fixed effects)
- For a given θ denote the choice probabilities for player i at time t and market m as $P_i(a_i^{mt}|x^{mt};\theta)$

$$\Big(P(\theta), V(\theta)\Big) \in \mathcal{E}(\Psi, \theta): \quad P(\theta) = \Big\{ P_i(a_i^{mt}|x^{mt};\theta) \Big\}_{i,m,t}$$

- MLE estimator $\hat{\theta}^{ML}$ is given by

$$\hat{\theta}^{ML} = \arg\max_{\theta} \left[\max_{(P(\theta),V(\theta)\in\mathcal{E}(\Psi,\theta)} \frac{1}{M} \sum_{i=1}^{N} \sum_{m=1}^{M} \sum_{t=1}^{T} \log P_i(\bar{a}_i{}^{mt}|\bar{x}^{mt};\theta) \right]$$

MLE via Constrained Optimization Approach

- Idea: use discretized values of P and V as *variables*
- Augmented log-likelihood function is

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

- The constrained optimization formulation of the ML estimation problem is

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

- Math programming with equilibrium constraints (MPEC)
- Does not rely *as much* on the structure of the problem
- Much bigger computational problem
- Implements the same MLE estimator (*when it works*)

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

Estimation methods for dynamic stochastic games

- Two step (CCP) estimators
  - Fast, do not impose equilibrium constraints, finite sample bias
  1. Estimate CCP $\to \hat{P}$
  2. Method of moments • Minimal distance • Pseudo likelihood

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

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

Aguirregabiria, Mira (2007); Aguirregabiria, Marcoux (2021)

- Efficient pseudo-likelihood (EPL)
  - Incorporates Newton step in the NPL operator
  - More robust to the stability and multiplicity of equilibria

Dearing, Blevins (2024), ReStud

Overview of NRLS

Full solution nested fixed point MLE estimator
with computational enhancements to ensure tractability

- 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

ROAD MAP

1. Solving directional dynamic games (DDGs):
   - Simple example: Bertrand pricing and investment game
   - State recursion algorithm
   - NRLS: NFXP using the Recursive lexicographical search (RLS) algorithm
2. Structural estimation of DDGs using Nested RLS
   - Branch-and-bound on RLS tree
   - Non-parametric likelihood bounding
3. Monte Carlo: (Compare NRLS, two-step CCP, NPL, EPL, MPEC)
   - One equilibrium in the model and data
   - Multiplicity of equilibria at true parameter
   - (Multiple equilibria in the data)

Amcor-Visy collusion case

- Australian market for cardboard is essentially a duopoly
- Between 2000 and 2005 *Visy* and *Amcor* colluded to divide the market of cardboard and to fix prices
- 2007: Visy admits to have been manipulating the market, issued with $36 million fine
- July 2009: Cadbury vs. Amcor, damages estimated at $235.8 million, settles out of court
- March 2011: Class action suit against both Amcor and Visy settles out of court for $95 million

Cardboard industry in Australia

- Cardboard is a highly standardized product
- Bertrand price competition with strong incentives for price cutting
- Amcor and Visy do minimal amounts of R&D themselves,
- Instead purchase new technology from other companies to reduce cost of production
  - Amcor plans to build state-of-the-art paper mill in Botany Bay before the collusion took place
  - "B9" plant finally opened on February 1, 2013

Leapfrogging equilibrium

- Firms invest in alternating fashion and take turns in cost leadership
- Market price makes permanent downward shifts

Dynamic Bertrand price competition

Directional stochastic dynamic game

- Two Bertrand competitors, $n = 2$, no entry or exit
- Discrete time, infinite horizon ($t = 1, 2, \dots, \infty$)
- 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 $\eta\epsilon_{i,N}$

Bellman equations, firm 1, simultaneous moves

$$V_1(c_1, c_2, c, \epsilon_1) = \max\left[v_1(I, c_1, c_2, c) + \eta\epsilon_1(I), v_1(N, c_1, c_2, c) + \eta\epsilon_1(N)\right]$$

$$\begin{aligned} v_1(N, c_1, c_2, c) &= r_1(c_1, c_2) + \beta EV_1(c_1, c_2, c, N) \\ v_1(I, c_1, c_2, c) &= r_1(c_1, c_2) - K(c) + \beta EV_1(c_1, c_2, c, I) \end{aligned}$$

With extreme value shocks, the investment probability (CCP) is

$$P_1(I|c_1, c_2, c) = \frac{\exp\{v_1(I, c_1, c_2, c)/\eta\}}{\exp\{v_1(I, c_1, c_2, c)/\eta\} + \exp\{v_1(N, c_1, c_2, c)/\eta\}}$$

- There is a separate Bellman equation for player 2, with "outputs" V_2 and P_2, where $P_2(I|c_1, c_2, c)$ is firm 2's probability of investing in state (c_1, c_2, c).

Bellman equations, firm 1, simultaneous moves

The expected values are given by

$$EV_1(c_1, c_2, c, N) =$$
$$\int_0^c \left[P_2(I|c_1, c_2, c) H_1(c_1, c, c') + [1 - P_2(I|c_1, c_2, c)] H_1(c_1, c_2, c') \right] \pi(dc'|c)$$

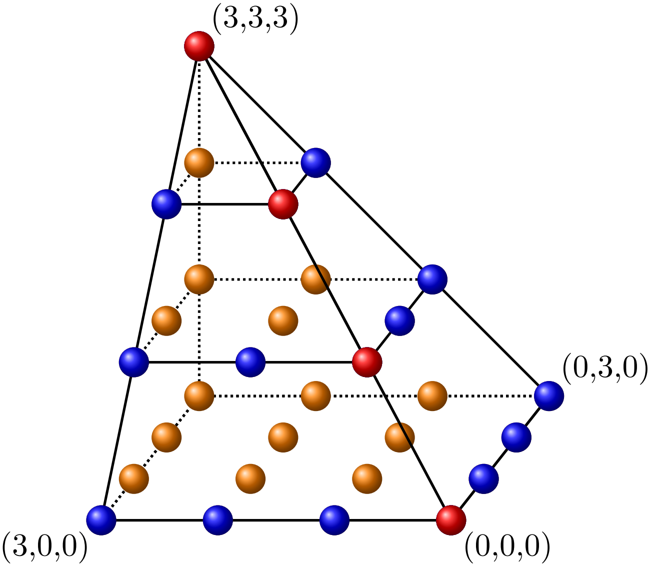
$$EV_1(c_1, c_2, c, I) =$$
$$\int_0^c \left[\quad P_2(I|c_1, c_2, c) H_1(c, c, c') + [1 - P_2(I|c_1, c_2, c)] H_1(c, c_2, c') \right] \pi(dc'|c)$$

$$H_1(c_1, c_2, c) = \eta \log \left[\exp\left(v_1^N(c_1, c_2, c)/\eta\right) + \exp\left(v_1^I(c_1, c_2, c)/\eta\right) \right].$$

is the "smoothed max" or logsum function

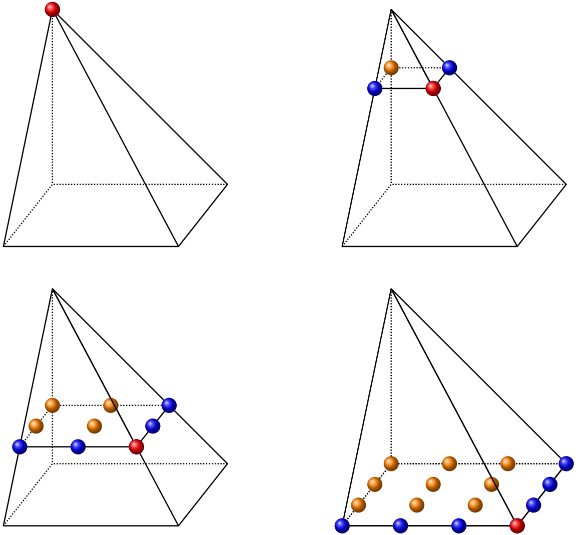
Discretized state space = a "quarter pyramid"

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



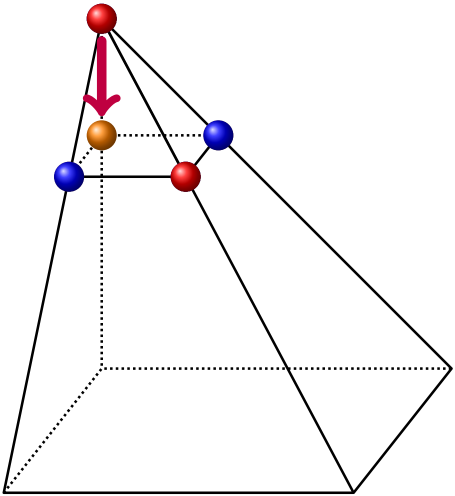
Transitions due to technological progress

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



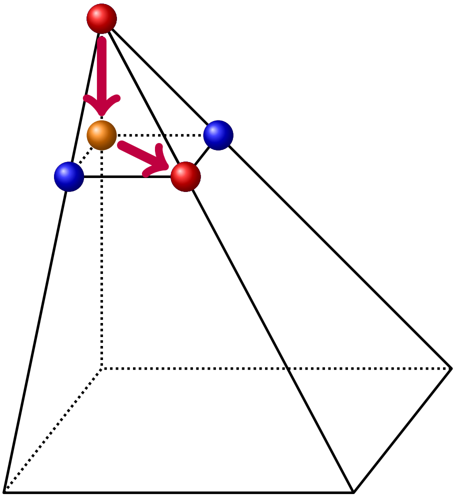
Game dynamics: example

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



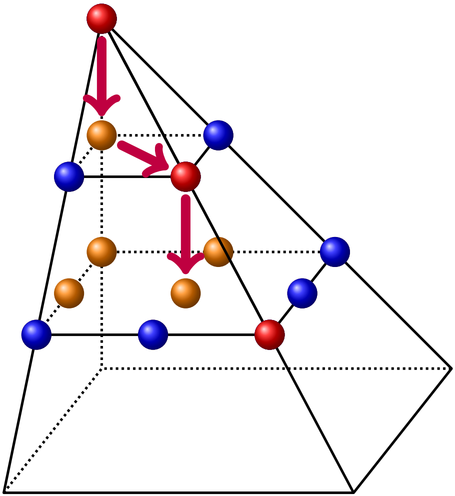
Game dynamics: example

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



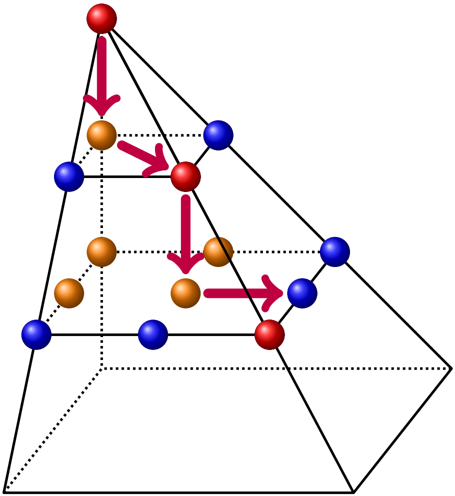
Game dynamics: example

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



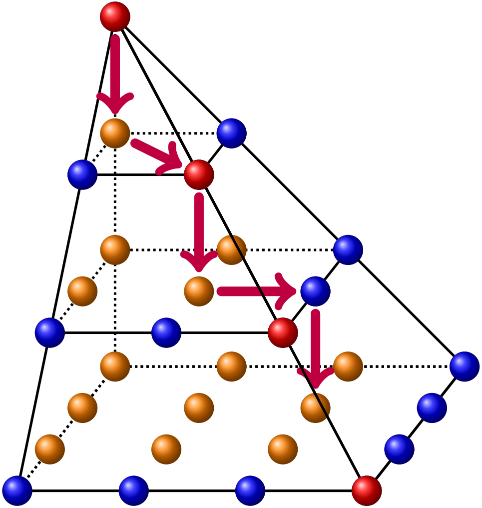
Game dynamics: example

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



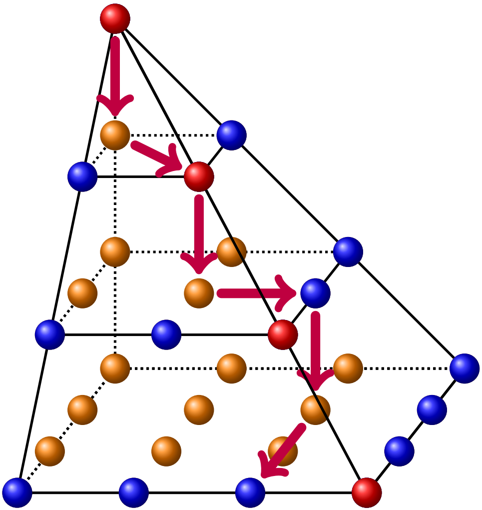
Game dynamics: example

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



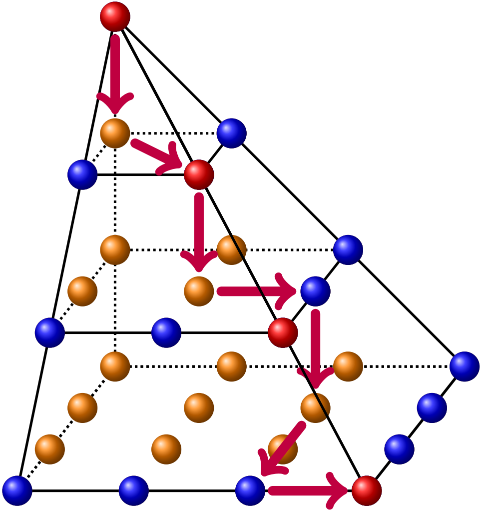
Game dynamics: example

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



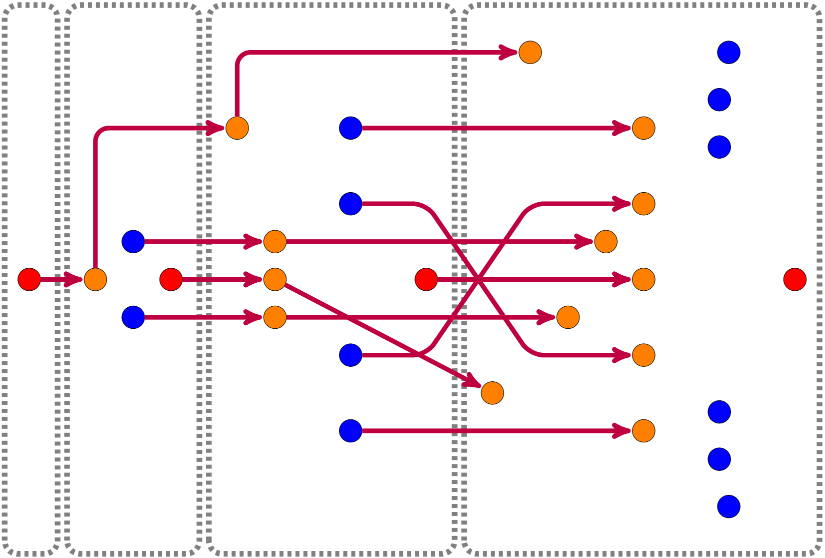
Game dynamics: example

A particular sequence of investment decisions along technological progress pass



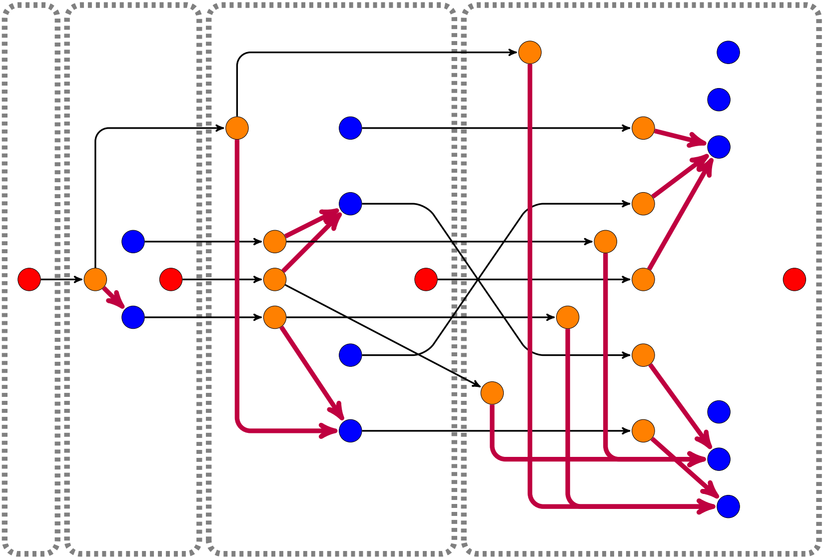
Transitions due to technological progress

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



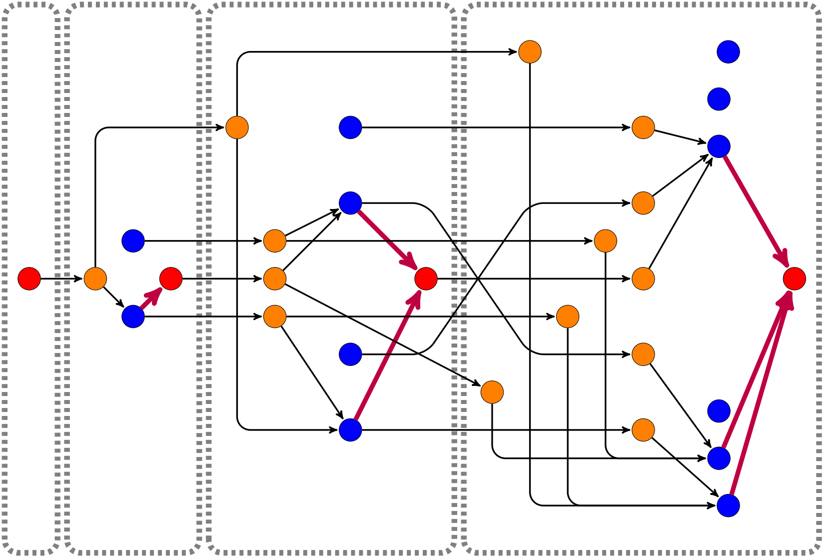
Strategy-specific partial order on S

Strategy σ_1 of firm 1: invest at all interior points



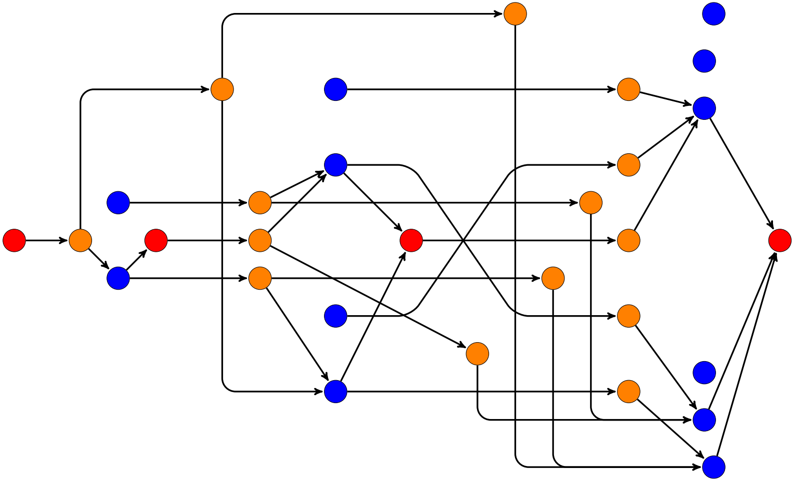
Strategy-specific partial order on S

Strategy σ_2 of firm 2: invest at all edge points



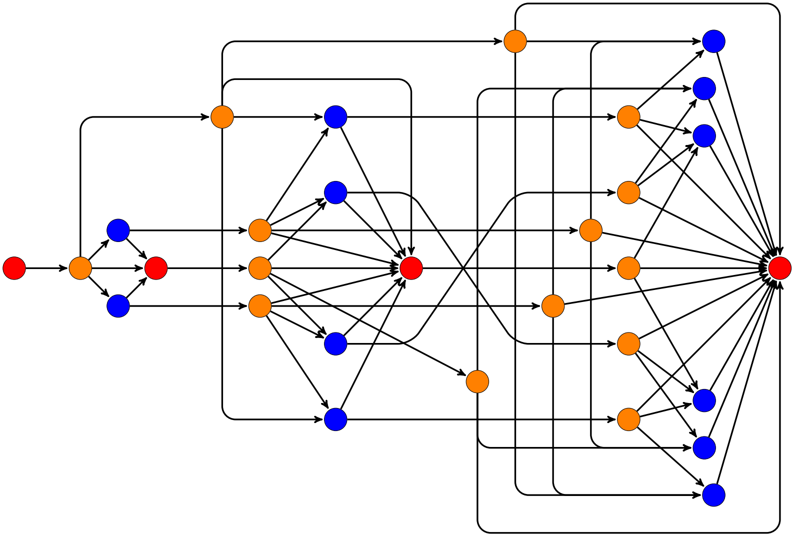
Strategy-specific partial order on S

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



Strategy independent partial order on S

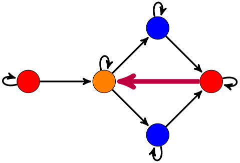
Coarsest common refinement of partial orders induced by all strategies



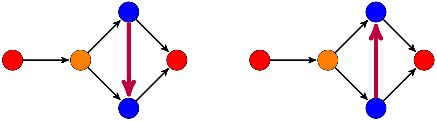
Definition of the Dynamic Directional Games

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

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



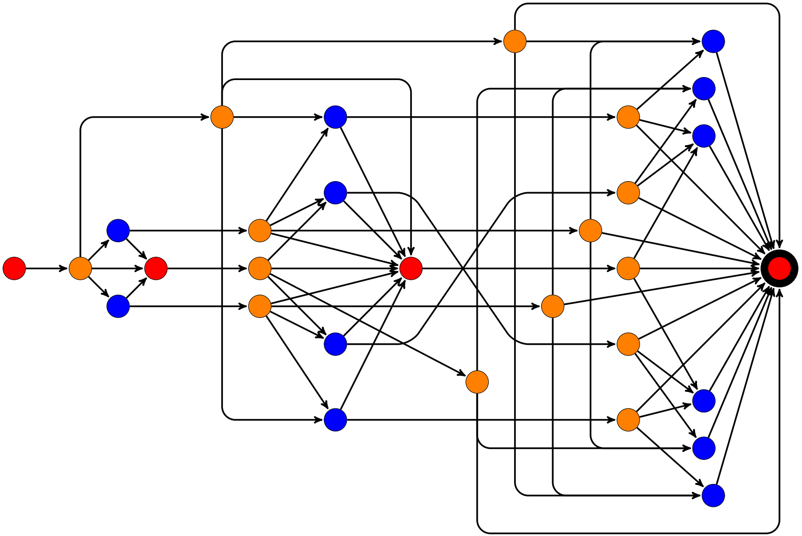
2. Every pair of feasible Markovian strategies σ and σ' induce consistent partial orders on the state space.



Iskhakov, Rust and Schjerning (2016)

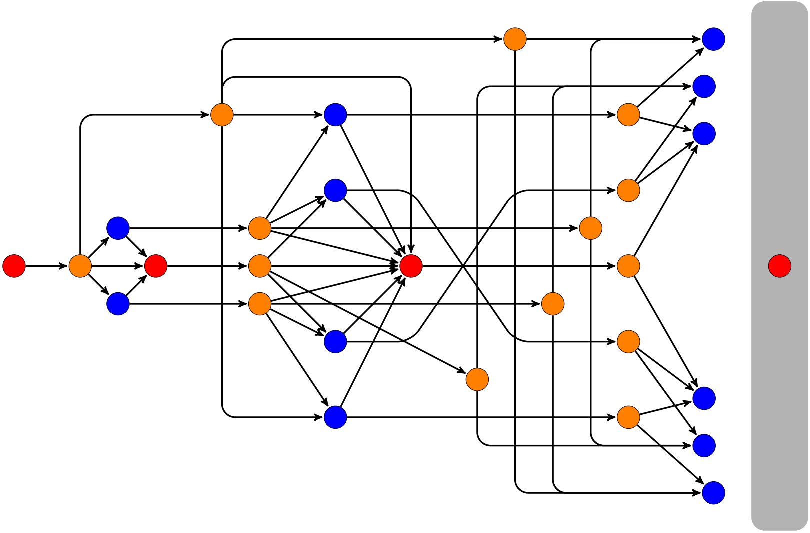
DAG recursion to partition S into stages

Identify terminal states



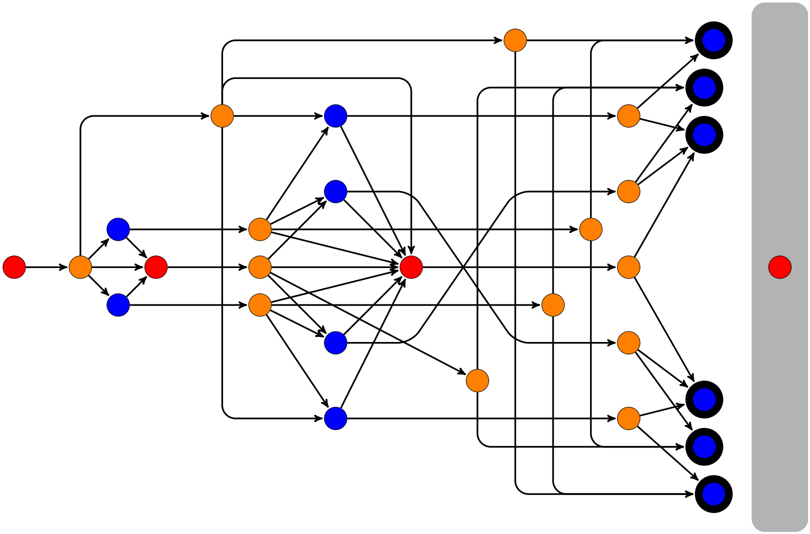
DAG recursion to partition S into stages

Remove terminal states



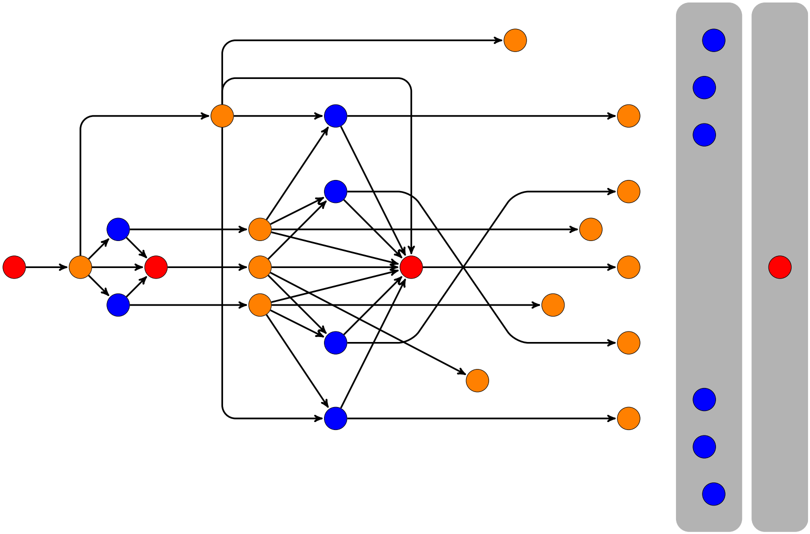
DAG recursion to partition S into stages

Identify terminal states



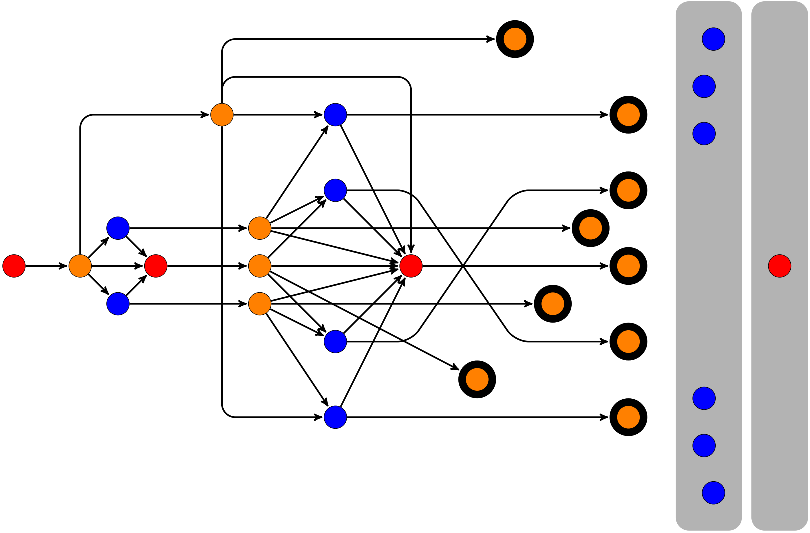
DAG recursion to partition S into stages

Remove terminal states



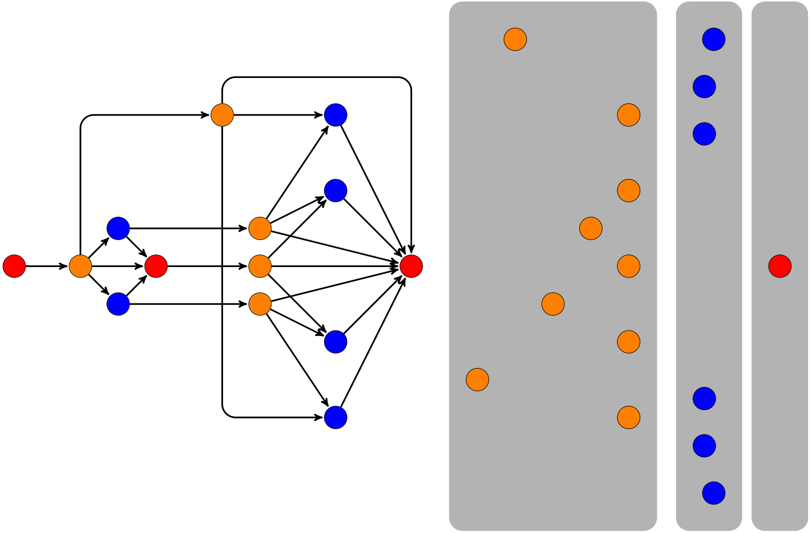
DAG recursion to partition S into stages

Identify terminal states



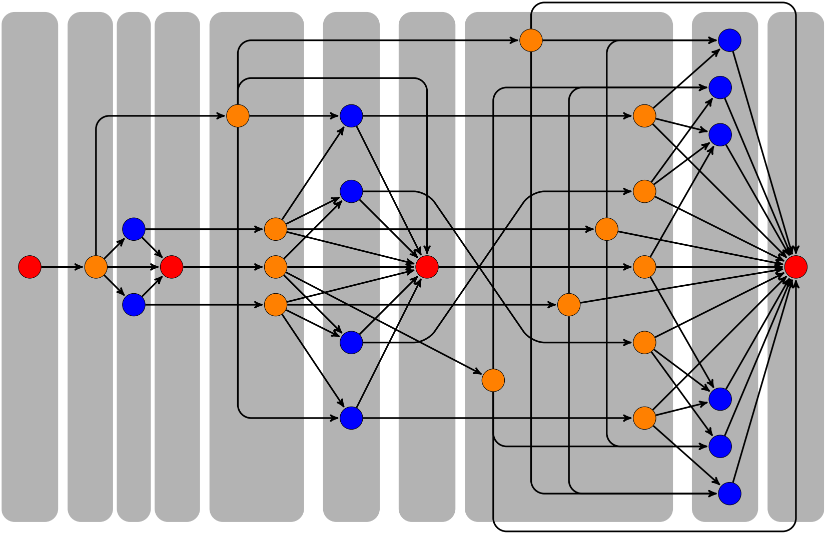
DAG recursion to partition S into stages

Remove terminal states



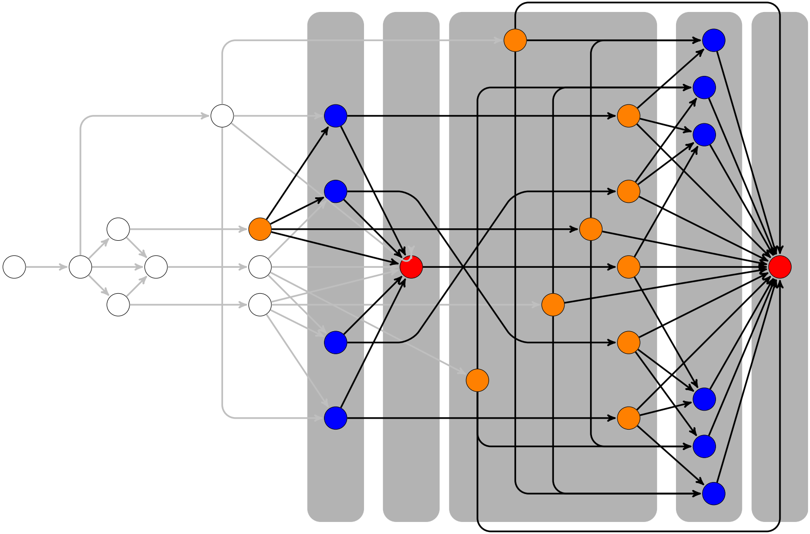
Total order on the set of stages

After running a topoligical sort algorithm on the DAG



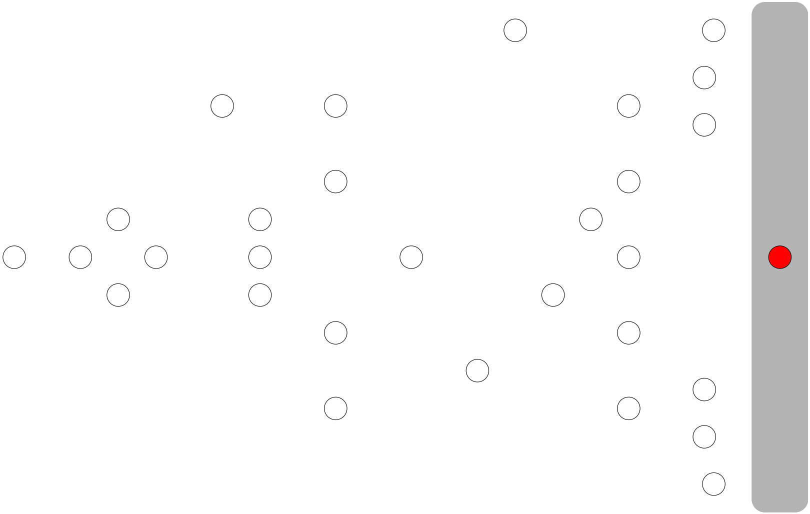
Subgames of DDG and continuation strategies

Subgames and continuation strategies



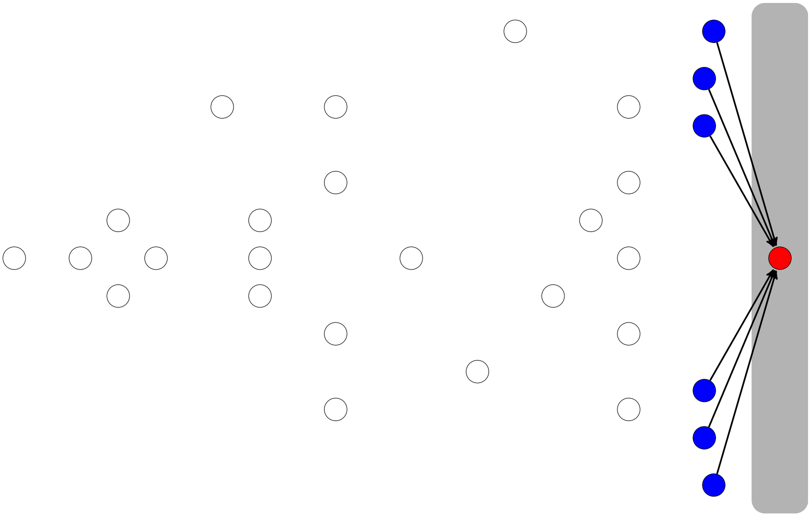
State recursion algorithm

Backward induction on stages of DDG



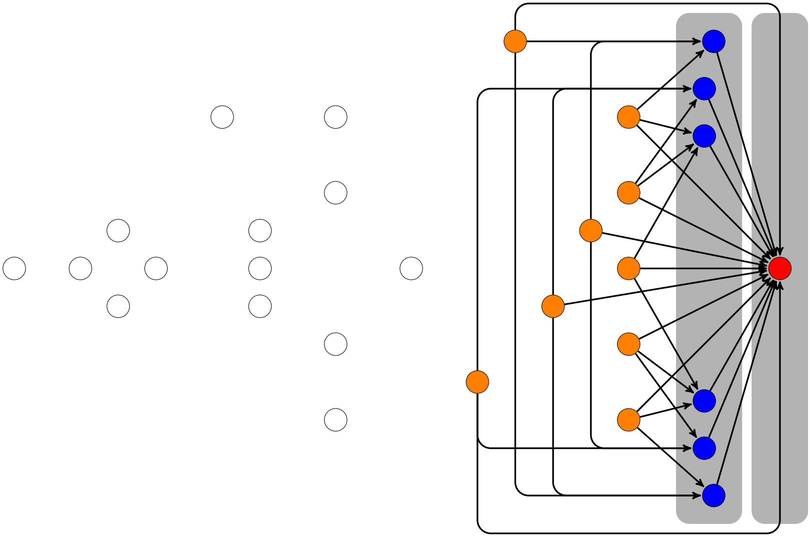
State recursion algorithm

Backward induction on stages of DDG



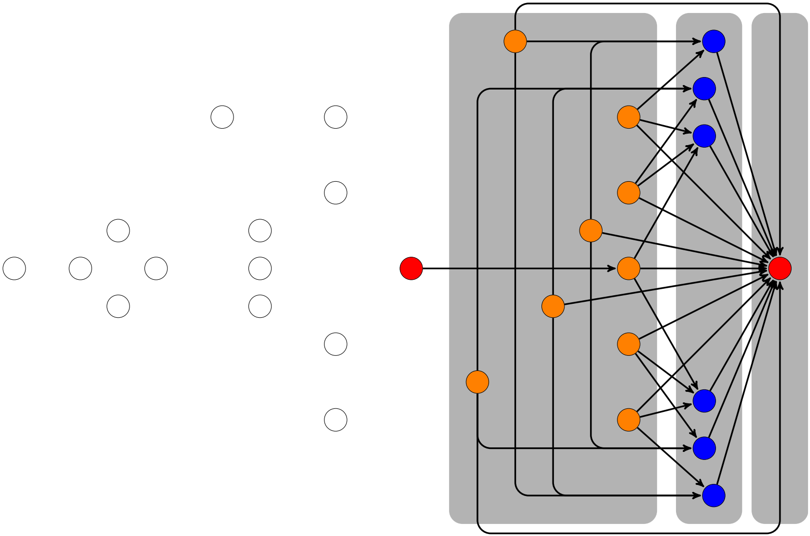
State recursion algorithm

Backward induction on stages of DDG



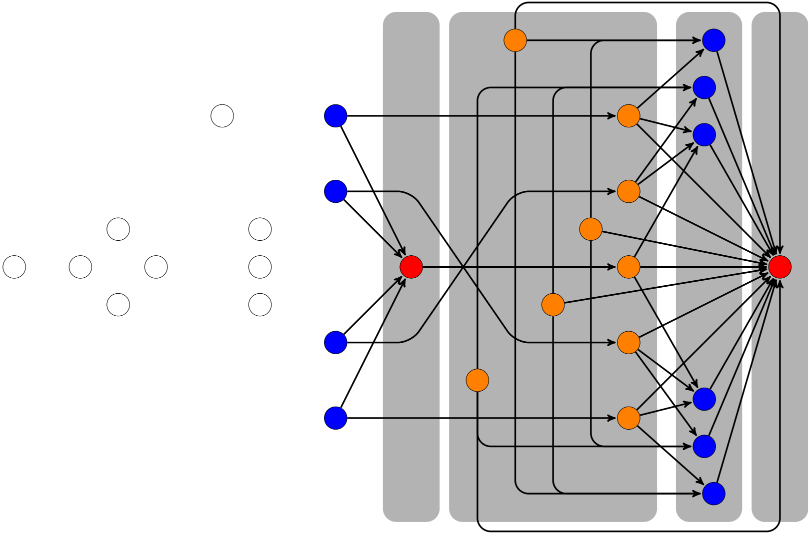
State recursion algorithm

Backward induction on stages of DDG



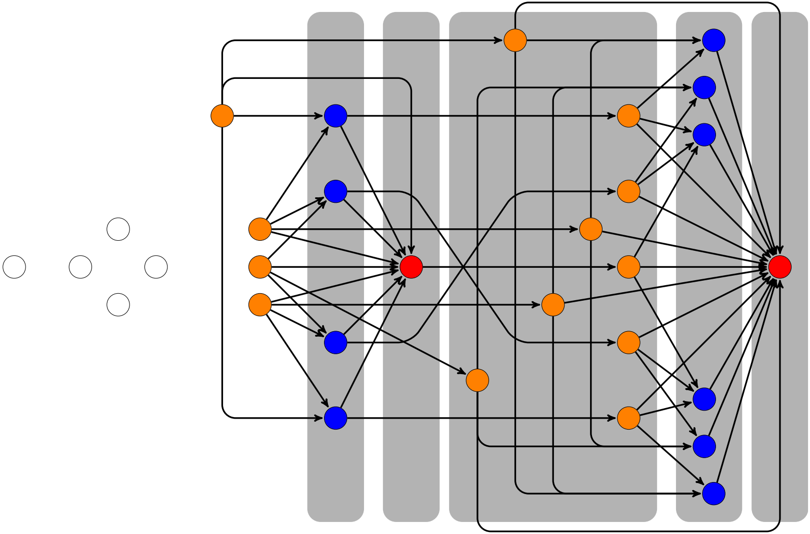
State recursion algorithm

Backward induction on stages of DDG



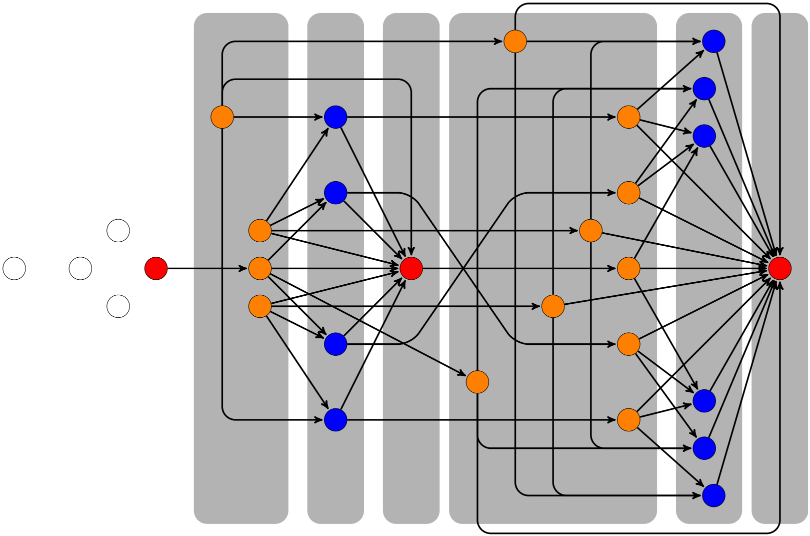
State recursion algorithm

Backward induction on stages of DDG



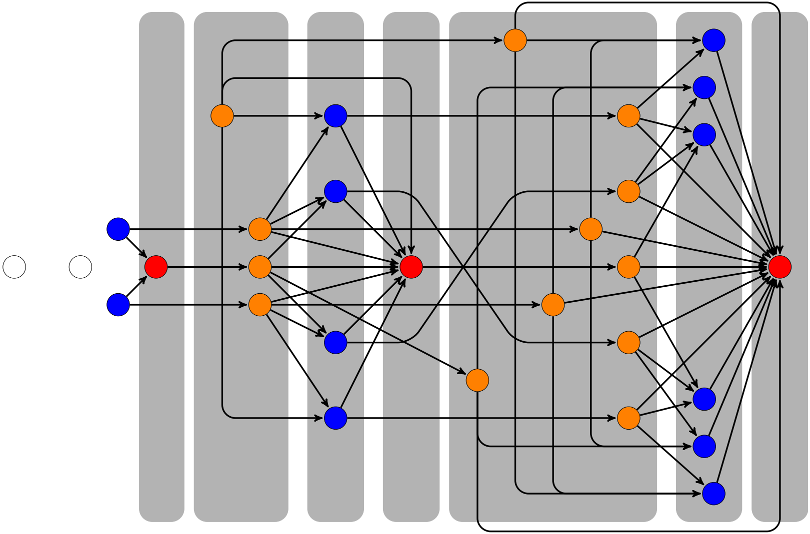
State recursion algorithm

Backward induction on stages of DDG



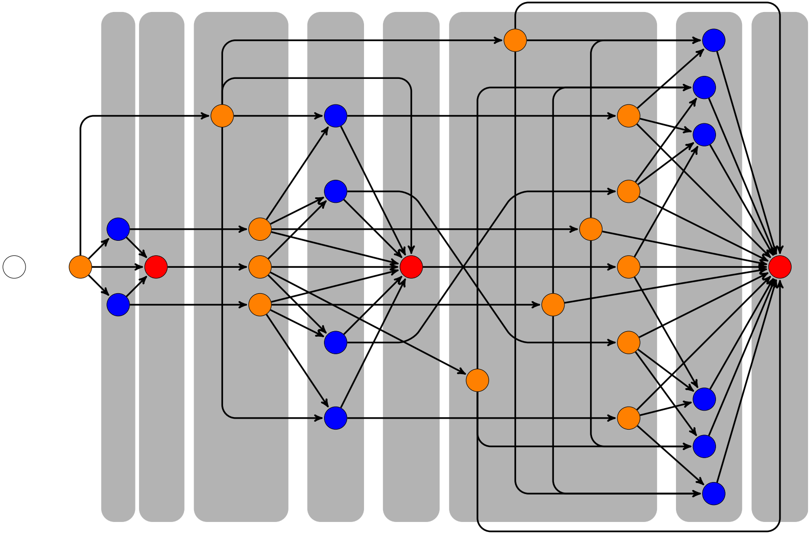
State recursion algorithm

Backward induction on stages of DDG



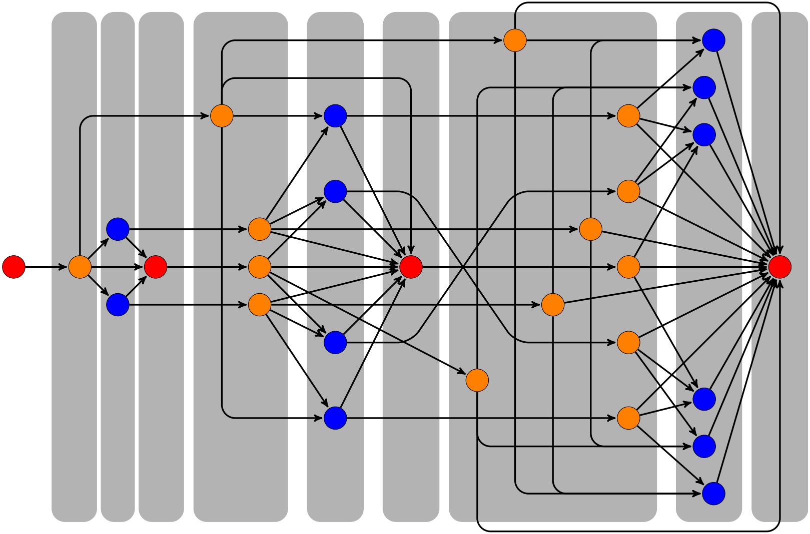
State recursion algorithm

Backward induction on stages of DDG



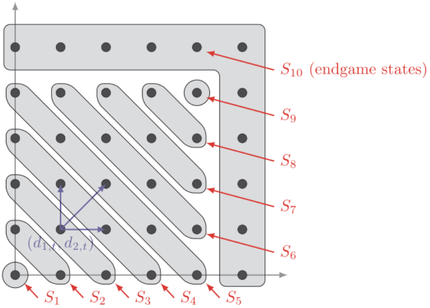
State recursion algorithm

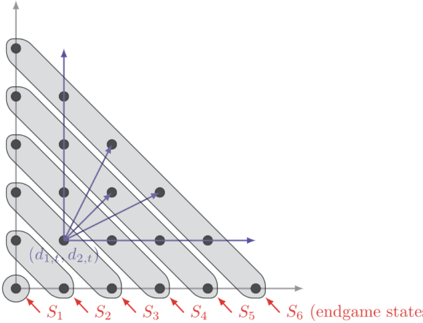
Backward induction on stages of DDG



Examples of Directional Dynamic Games

Many games have state dynamic evolutions described by a DAGs

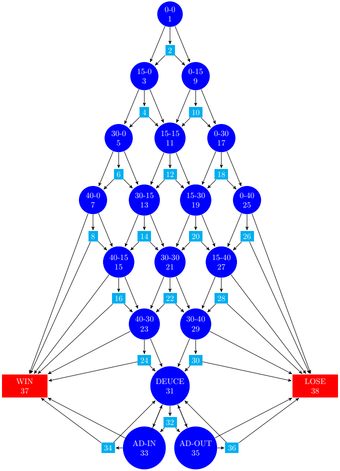




Judd, Schmedders, Yeltekin (2012), *IER*
"Optimal rules for patent researchers"

Dube, Hitsch, Chintagunta (2010), *Marketing Science*
"Tipping and concentration in markets with indirect network effects"

Tennis is a Directional Dynamic Game

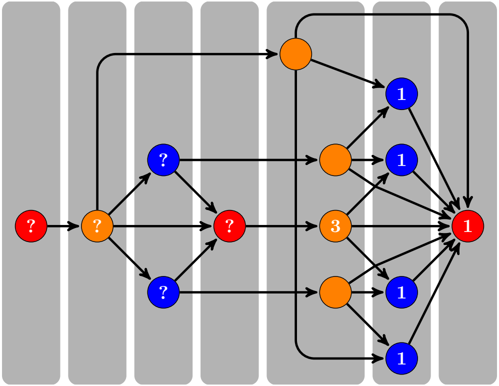


Anderson, Rosen, Rust, Wong (2024) *JPE*
"Disequilibrium Play in Tennis"

Multiplicity of stage equibiria

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

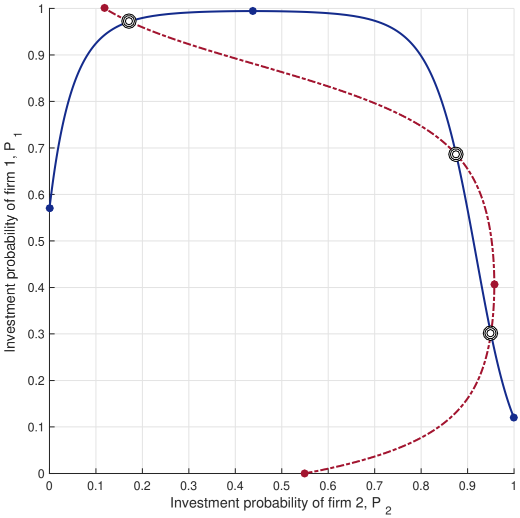
- State recursion proceeds conditional on equilibrium selection rule
- Multiplicity of stage equilibria ⇔ multiplicity
- Can systematically combine different stage equilibria



Best response functions

Typically one or three stage equilibria, but may be 5

- Smooth best response function with $\eta > 0$



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

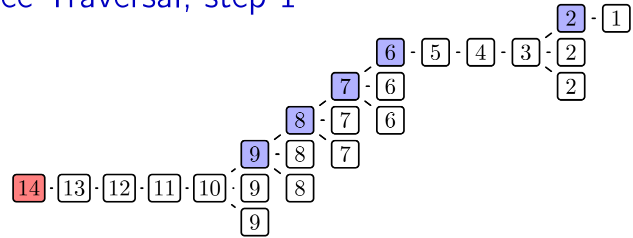
Challenge:

- 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

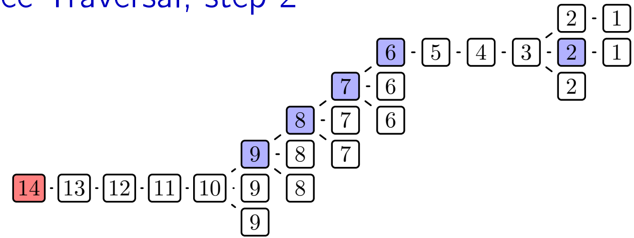
Solution: RLS = depth-first tree traversal (illustration coming)

- Root of the tree is one of the absorbing states
- Levels of the tree correspond to the state points
- Branching happens when stages have multiple equilibria
- MPE of the game is given by a path from root to a leaf

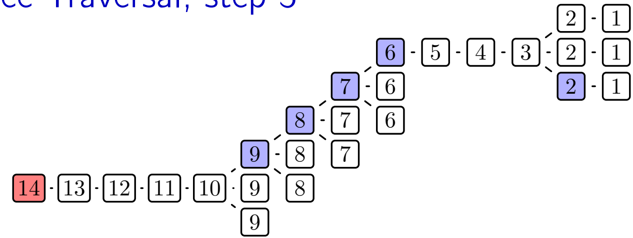
RLS Tree Traversal, step 1



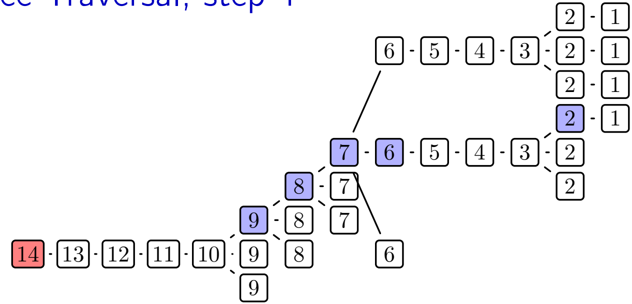
RLS Tree Traversal, step 2



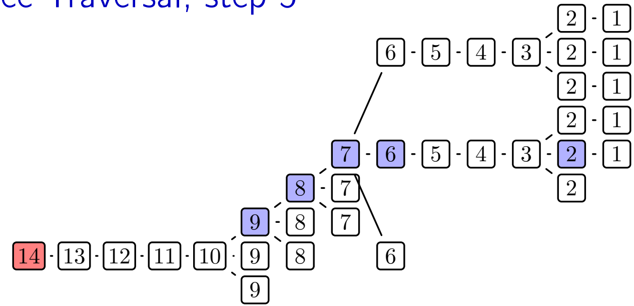
RLS Tree Traversal, step 3



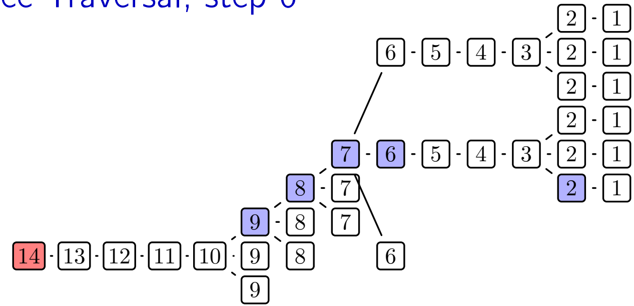
RLS Tree Traversal, step 4



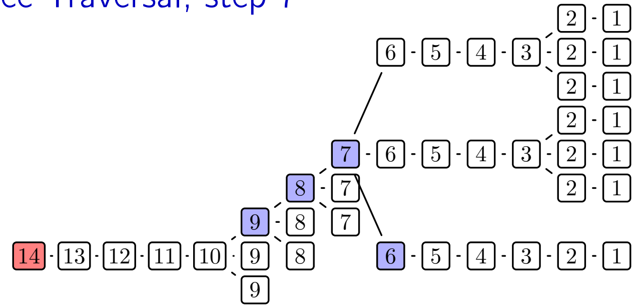
RLS Tree Traversal, step 5



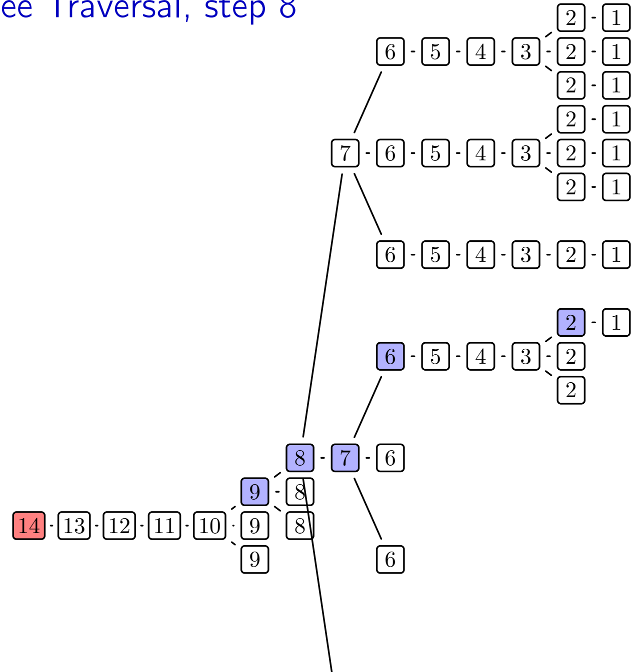
RLS Tree Traversal, step 6



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
   - Branch-and-bound on RLS tree
   - Non-parametric likelihood bounding
3. Monte Carlo: (Compare NRLS, two-step CCP, NPL, EPL, MPEC)
   - One equilibrium in the model and data
   - Multiplicity of equilibria at true parameter
   - (Multiple equilibria in the data)

Nested Recursive Lexicographical Search (NRLS)

- Data from M independent markets from T periods
  $Z = \left\{a^{jt}, x^{jt}\right\}_{j\in\{1,\dots,N\},t\in\{1,\dots,T\}}$
- Let the set of all MPE equilibria be $\mathcal{E} = \{1, \dots, K(\theta)\}$

1. Outer loop
   Maximization of the likelihood function w.r.t. to structural parameters θ
   $$\theta^{ML} = \arg\max_{\theta\in\Theta} \mathcal{L}(Z, \theta)$$
2. Inner loop
   Maximization of the likelihood function w.r.t. equilibrium selection
   $$\mathcal{L}(Z, \theta) = \arg\max_{k\in\{1,\dots,K(\theta)\}} \mathcal{L}(Z, \theta, V_\theta^k)$$

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

Likelihood over the state space

- Given equilibrium k choice probabilities $P_i^k(a|x)$, likelihood is

$$\mathcal{L}(Z,\theta,V_\theta^k) = \sum_{j=1}^{N}\sum_{t=1}^{T}\sum_{i=1}^{J}\log P_i^k(a_i^{jt}|x^{jt};\theta)$$

- Let ι index points in the state space
  $\iota = 1$ initial point, $\iota = S$ the terminal state
- Denote n_ι the number of observations in state x_ι and $n_\iota^{a_i}$ the number of observations of player i taking action a_i at x_ι

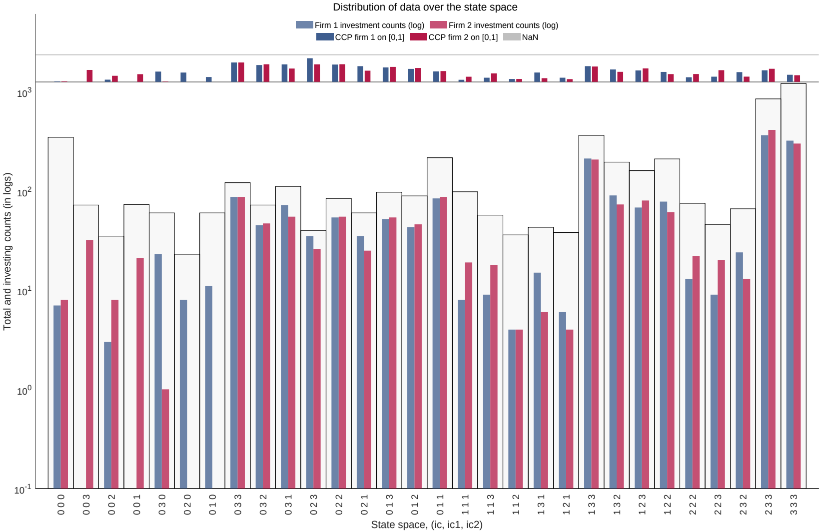
$$n_\iota = \sum_{j=1}^{N}\sum_{t=1}^{T}\mathbb{1}\{x^{jt} = x_\iota\} \qquad n_\iota^{a_i} = \sum_{j=1}^{N}\sum_{t=1}^{T}\mathbb{1}\{a_i^{jt} = a_i, x^{jt} = x_\iota\}$$

- Then equilibrium-specific likelihood can be computed as

$$\mathcal{L}(Z,\theta,V_\theta^k) = \sum_{\iota=1}^{S}\sum_{i=1}^{J}\sum_{a} n_\iota^{a_i}\log P_i^k(a|x_\iota;\theta)$$

Data distribution over the state space

1000 markets, 5 time periods, init at apex of the pyramid



Branch and bound (BnB) method

Land and Doig, 1960 *Econometrica*

- Old method for solving integer programming problems

- **Branching**: RLS tree
- **Bounding**: The bound function is partial likelihood of equilibrium k calculated on the subset of states $\iota \in \mathcal{S} \subset \{1, \dots, S\}$

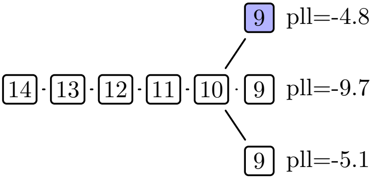
$$\mathcal{L}^{\text{part}}(Z^{\mathcal{S}}, \theta, V_\theta^k) = \sum_{\iota \in \mathcal{S}} \sum_{i=1}^{J} \sum_{a} n_\iota^{a_i} \log P_i^k(a|x_\iota; \theta)$$

- Where $Z^{\mathcal{S}} = \{(a, x) : x \in \mathcal{S}\}$ denotes data observed on $\mathcal{S}$
- Monotonic decreasing in cardinality of $\mathcal{S}$ (declines as more data is added)
- Equals to the full log-likelihood on the full state space when $Z^{\mathcal{S}} = Z$ (at the leafs of RLS tree)

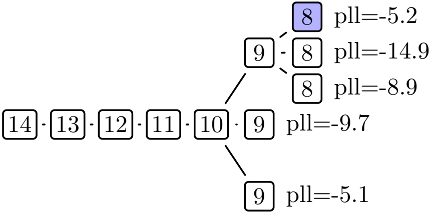
BnB on RLS tree, step 1

14 · 13 · 12 · 11 · 10 Partial loglikelihood = -3.2

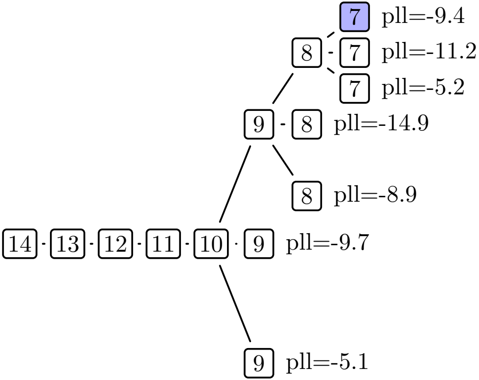
BnB on RLS tree, step 2



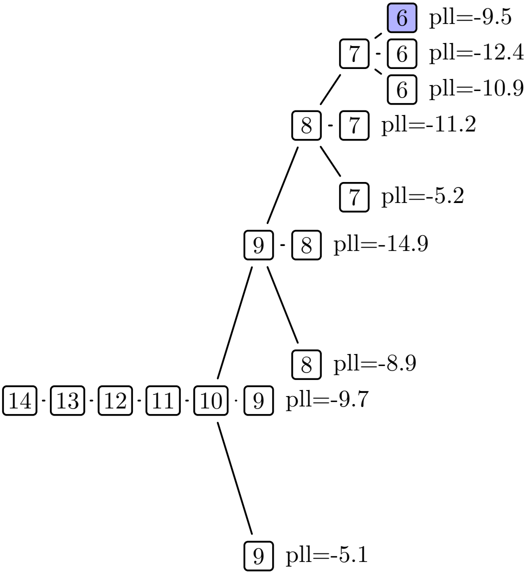
BnB on RLS tree, step 3



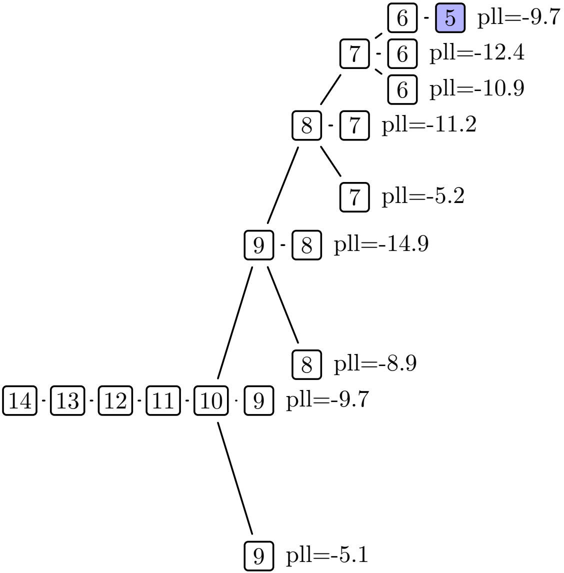
BnB on RLS tree, step 4



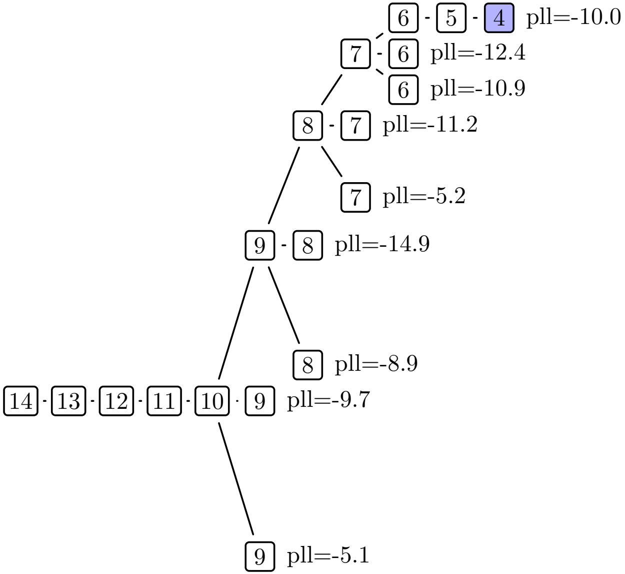
BnB on RLS tree, step 5



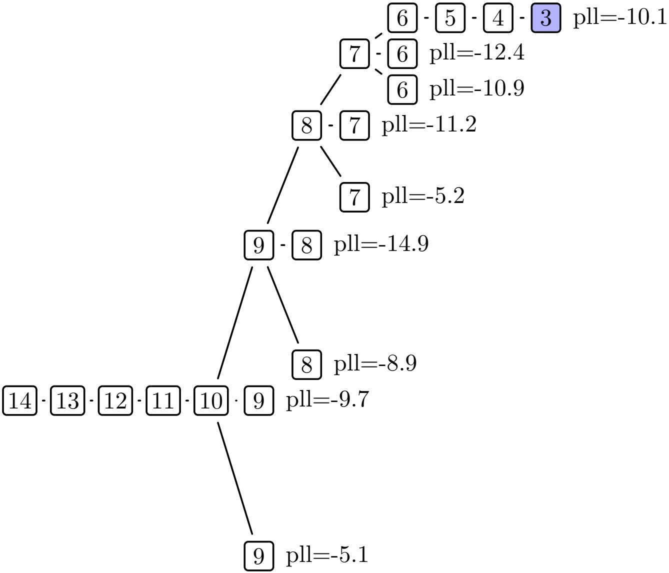
BnB on RLS tree, step 6



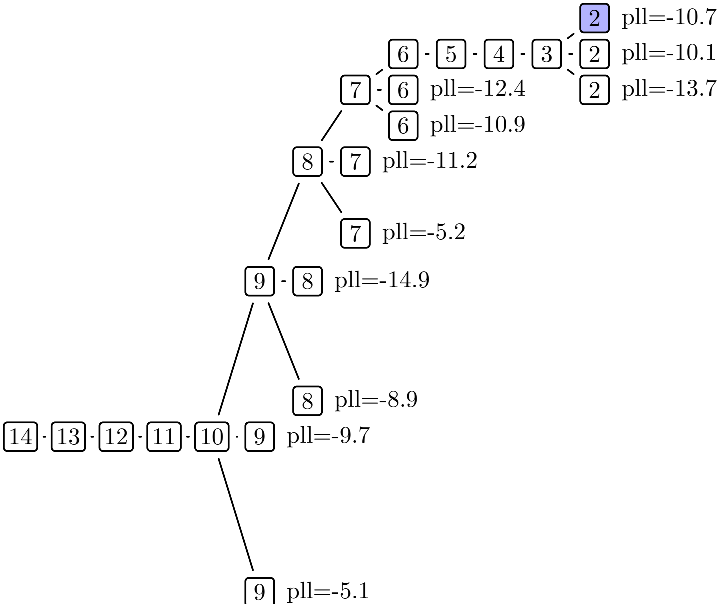
BnB on RLS tree, step 7



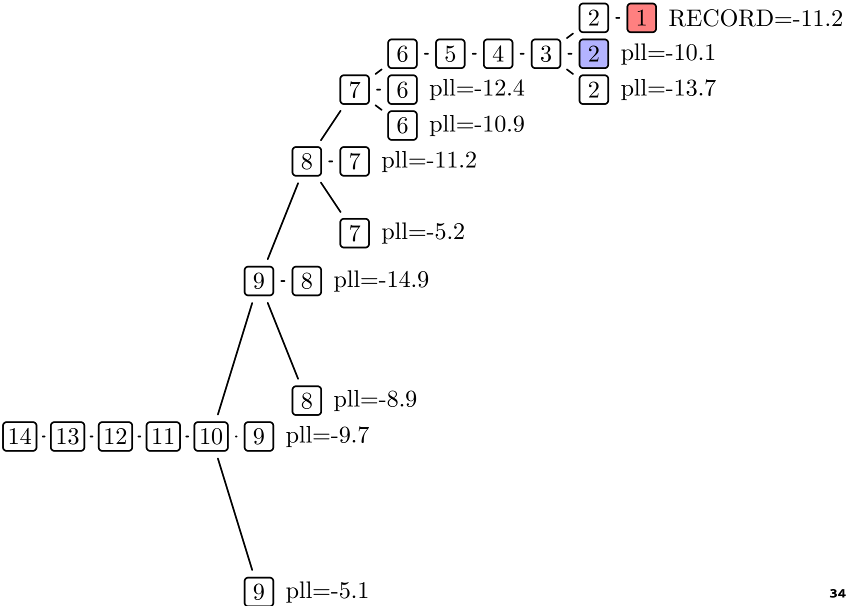
BnB on RLS tree, step 8



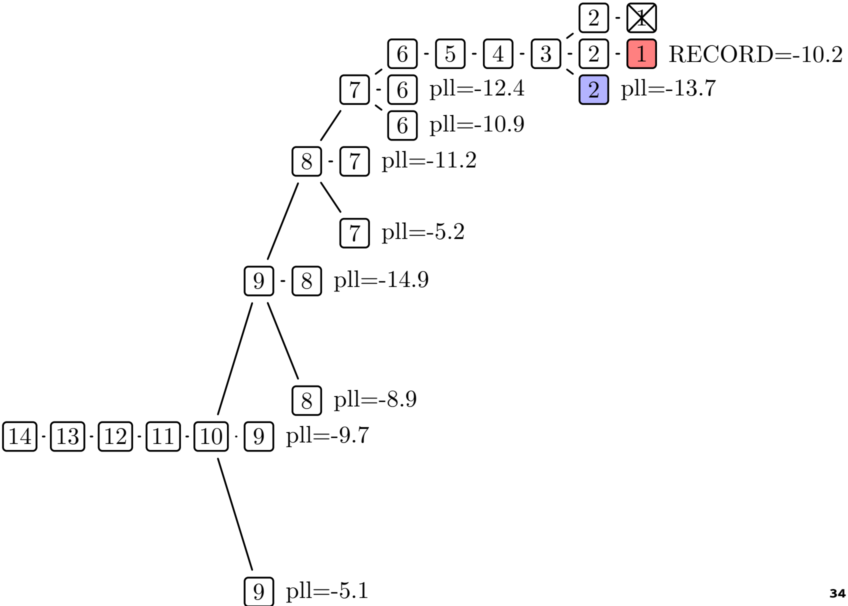
BnB on RLS tree, step 9



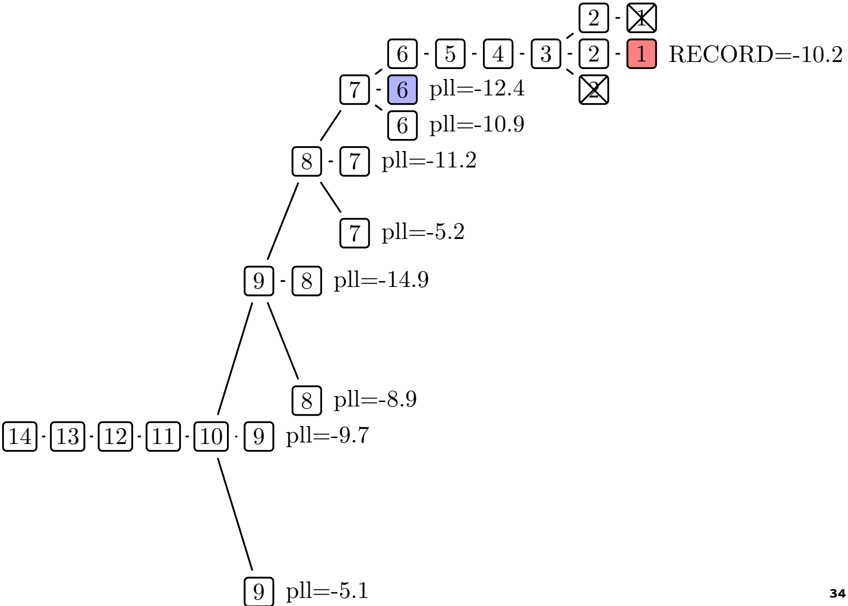
BnB on RLS tree, step 10



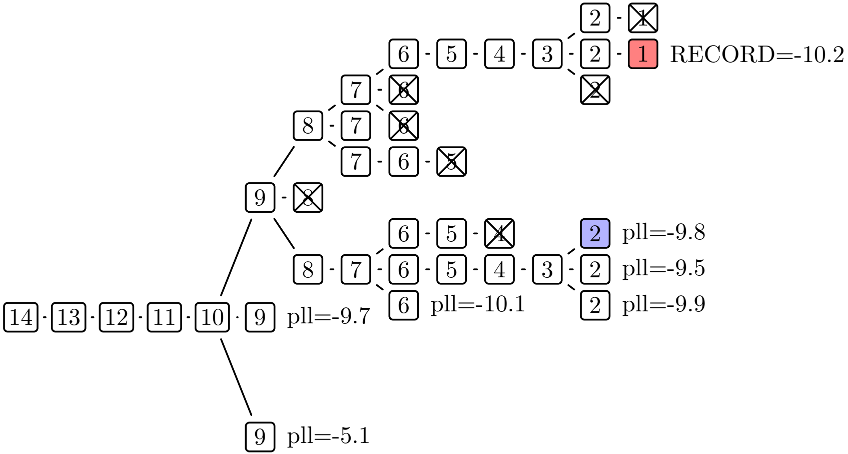
BnB on RLS tree, step 11



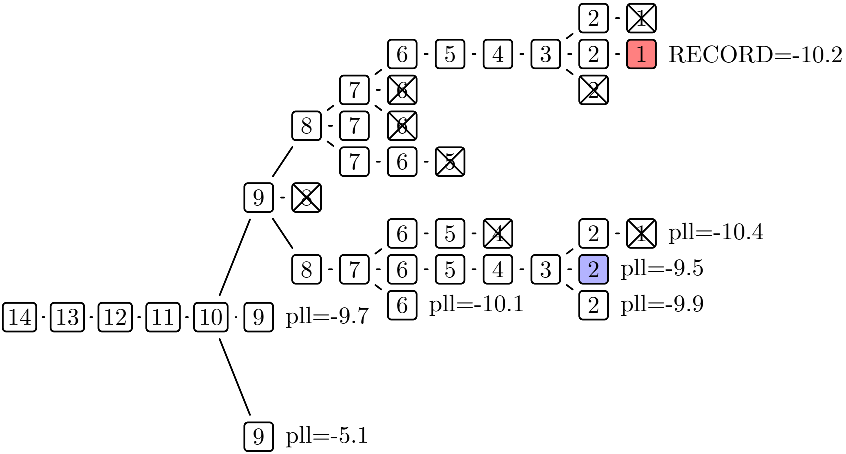
BnB on RLS tree, step 12



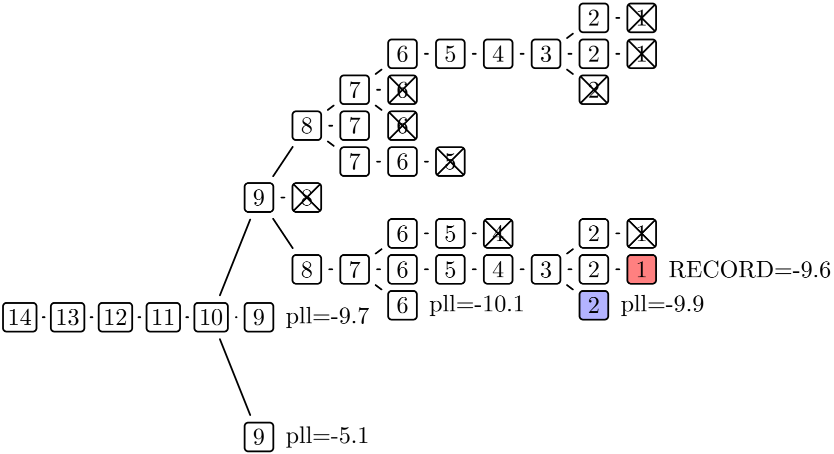
BnB on RLS tree, step 28



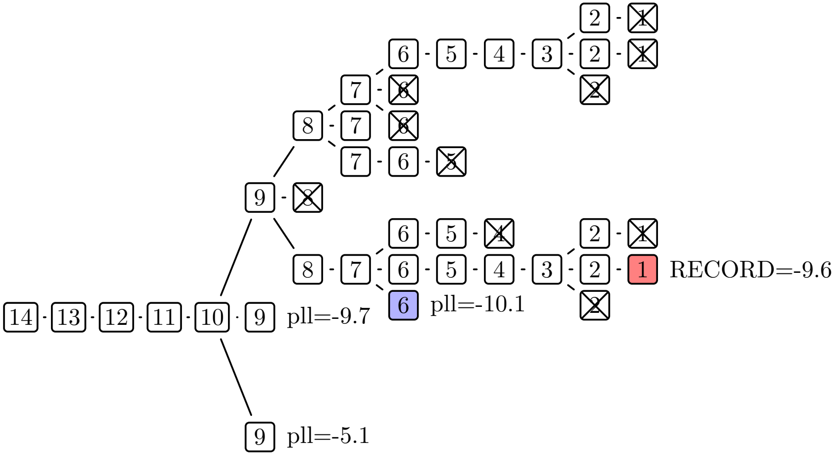
BnB on RLS tree, step 29



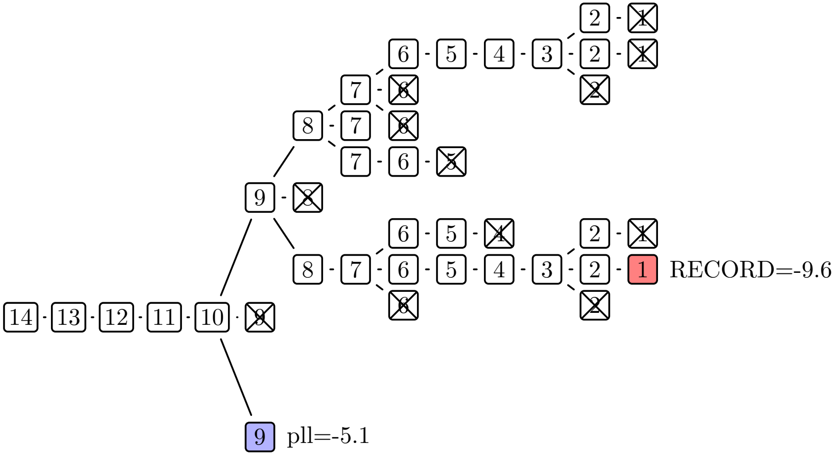
BnB on RLS tree, step 30



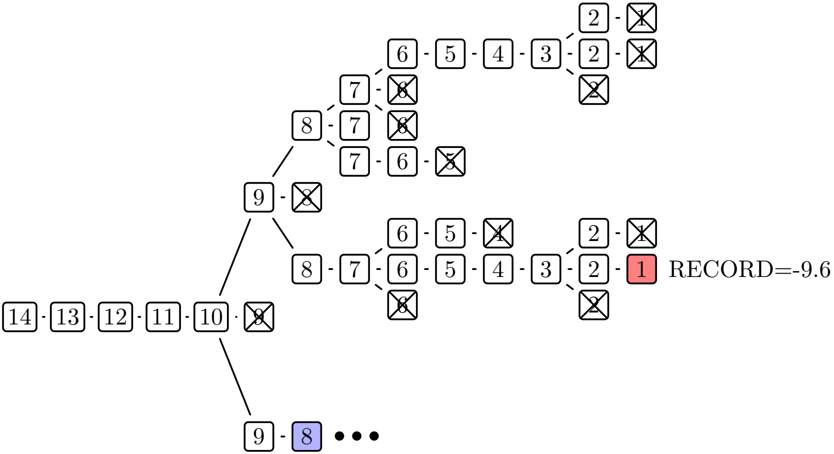
BnB on RLS tree, step 31



BnB on RLS tree, step 33



BnB on RLS tree, step 34



Non-parametric likelihood bounding

- Replace choice probabilities $P_i^k(a|x_\iota;\theta)$ with frequencies n_ι^a/n_ι

$$\mathcal{L}^{\text{non-par}}(Z^{\mathcal{S}}) = \sum_{\iota\in\mathcal{S}} \sum_{i=1}^{J} \sum_{a} n_\iota^{a_i} \log(n_\iota^a/n_\iota)$$

- $\mathcal{L}^{\text{non-par}}(Z^{\mathcal{S}})$ depends only on the counts from the data!
- Not hard to show algebraically that for any $Z^{\mathcal{S}}$ (≈Gibbs inequality)

$$\mathcal{L}^{\text{non-par}}(Z^{\mathcal{S}}) > L^{\text{part}}(Z^{\mathcal{S}}, \theta, V_\theta^k)$$

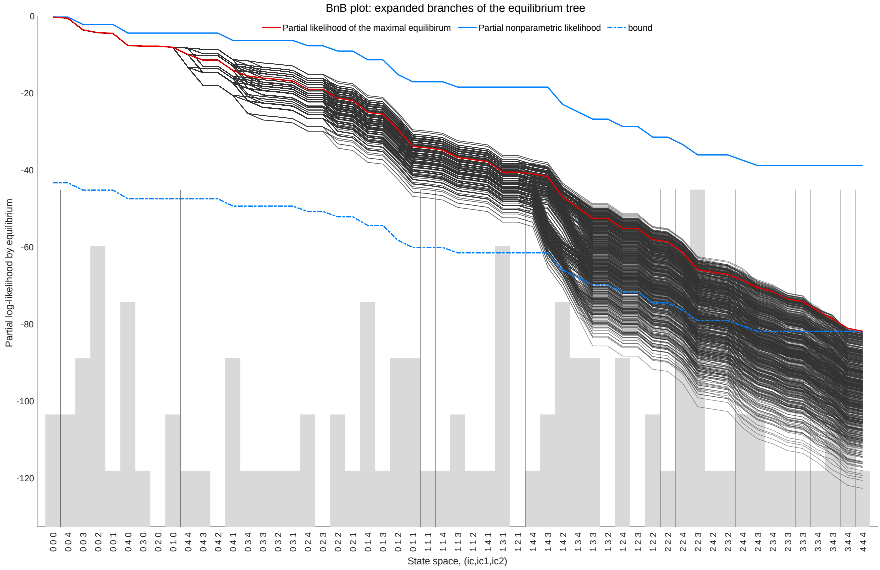
- Therefore partial likelihood can be optimistically extrapolated by empirical likelihood at any step ι of the RLS tree traversal

$$\mathcal{L}^{\text{part}}(Z^{\{S,S-1,\dots,\iota\}}, \theta, V_\theta^k) + \mathcal{L}^{\text{non-par}}(Z^{\{\iota-1,\dots,1\}})$$

- Augmented partial likelihood is much more powerful bound for BnB

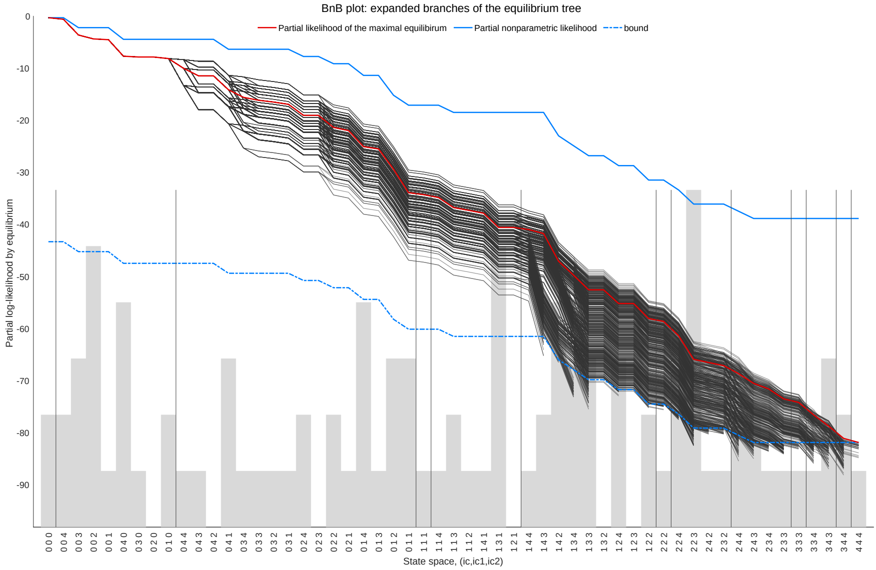
Non-parameteric likelihood bounding

$\iota = S = 14$ (terminal state) on the left, $\iota = 1$ (initial state) on the right



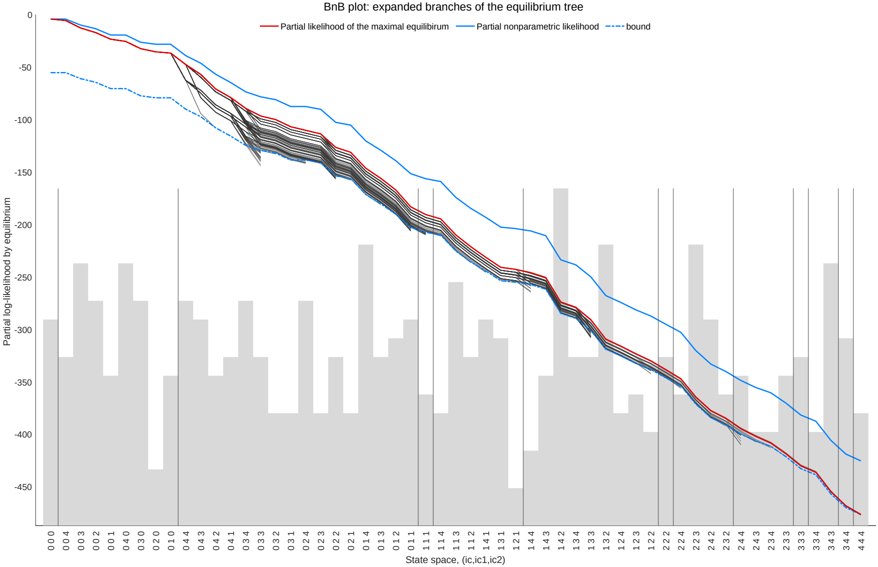
BnB with non-parameteric likelihood bound

Greedy traversal + non-parameteric likelihood bound



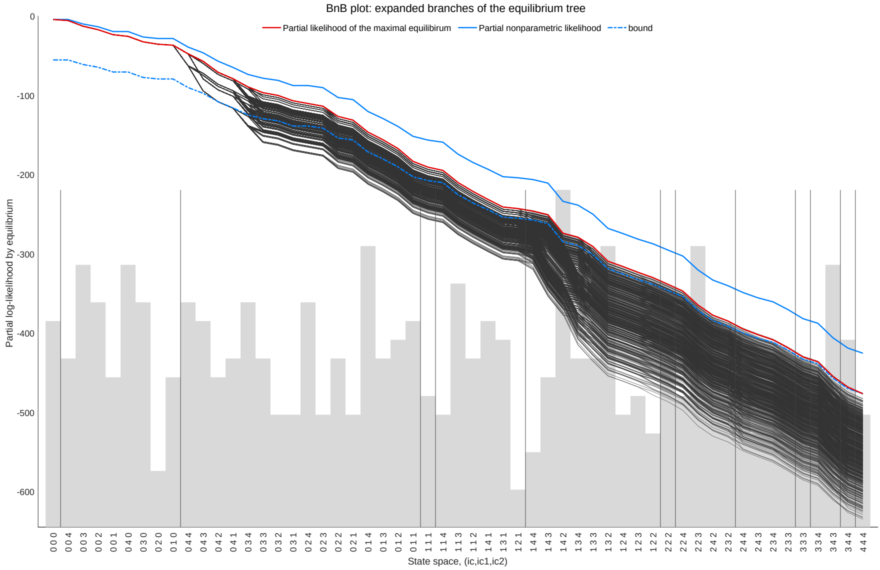
BnB with non-parameteric likelihood bound, larger sample

Non-parametric $\rightarrow$ parametric likelihood as $N \rightarrow \infty$ at true $\theta \Rightarrow$ even less computation



Full enumeration RLS in larger sample

Comparing with the previous slide most of the computation is avoided!



BnB refinement with non-parametric likelihood

- For any amount of data the non-parametric likelihood is greater or equal to the parametric likelihood *algebraically*
- BnB augmented with non-parameteric likelihood bound gives sharper Bounding Rules $\rightarrow$ less computation

- Wih more data as $M \rightarrow \infty$
- Non-parametric log-likelihood converge to the likelihood line
- The width of the band between the blue lines in the plots decreases

  $\rightarrow$ Even sharper Bounding Rules
  $\rightarrow$ Even less computation

MLE for any sample size, but easier to compute with more data!

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
   - Branch-and-bound on RLS tree
   - Non-parametric likelihood bounding
3. Monte Carlo: (Compare NRLS, two-step CCP, NPL, EPL, MPEC)
   - One equilibrium in the model and data
   - Multiplicity of equilibria at true parameter
   - (Multiple equilibria in the data)

Monte Carlo simulations

A

Single equilibrium in the model
One equilibrium in the data

B

Multiple equilibria in the model
Same equilibrium played the data

C

Multiple equilibria in the model
Multiple equilibria in the data:

- Long panels, each market plays their own equilibrium
- Groups of markets play the same equilibrium

(*not today*)

Implementation details

- Two-step estimator, NPL and EPL
  - Matlab unconstrained optimizer (with numerical derivatives)
  - CCPs from frequency estimators
  - Max 120 iterations (for NPL and EPL)
- MPEC
  - Matlab constraint optimizer (interior-point) with analytic derivatives
  - 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 (twice less variables)
  - Starting values from two-step estimator
- Estimated parameter k_1
- Sample size: 1000 markets in 5 time periods
- Parameters are chosen to ensure good coverage of the state space and non-degenerate CCPs in all states

Monte Carlo A, run 1: no multiplicity

Number of equilibria at true parameter: 1
Number of equilibria in the data: 1

	2step	NPL	EPL	MPEC-VP	MPEC-P	NRLS
True $k_1 = 3.5$	3.52786	3.49714	3.49488	3.49488	3.49486	3.49488
Bias	0.02786	-0.00286	-0.00512	-0.00512	-0.00514	-0.00512
MCSD	0.10037	0.06522	0.07042	0.07042	0.07078	0.07042
ave log-like	-1.16661	-1.16144	-1.16143	-1.16143	-1.16139	-1.16143
log-likelihood	-5833.07	-5807.21	-5807.16	-5807.16	-5806.95	-5807.16
log-like short	-	-0.050	-0.000	-0.000	-0.000	-0.000
KL divergence	0.03254	0.00021	0.00024	0.00024	0.00024	0.00024
$\|\|P - P0\|\|$	0.11270	0.00469	0.00495	0.00495	0.00500	0.00495
$\|\|\Psi(P) - P\|\|$	0.16185	0.0000	0.0000	0.0000	0.0000	0.0000
$\|\|\Gamma(v) - v\|\|$	0.87095	0.00000	0.00000	0.00000	0.00000	0.00000
Convrged of 100	-	100	100	100	99	100

- Equilibrium conditions satisfied (except 2step)
- Nearly all MLE estimators identical to the last digit
- NPL and EPL estimators approach MLE

Monte Carlo B, run 1: little multiplicity

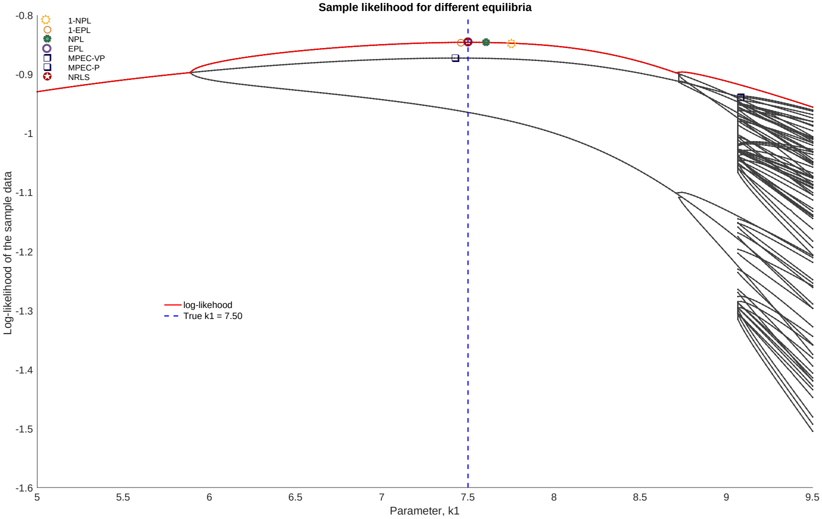
Number of equilibria at true parameter: 3
Number of equilibria in the data: 1
Data generating equilibrium: stable

	2step	NPL	EPL	MPEC-VP	MPEC-P	NRLS
True k1=7.5	7.55163	7.49844	7.49918	7.65318	7.35124	7.49919
Bias	0.05163	-0.00156	-0.00082	0.15318	-0.14876	-0.00081
MCSD	0.17875	0.06062	0.03413	0.99742	0.47136	0.03413
ave log-like	-0.84779	-0.84425	-0.84421	-0.88682	-0.87541	-0.84421
log-likelihood	-21194.86	-21106.33	-21105.13	-22170.40	-21885.37	-21105.13
log-like short	-	-1.206	-0.000	-1062.740	-776.809	-0.000
KL divergence	0.02557	0.00040	0.00013	0.23536	0.16051	0.00013
$\|\|P - P_0\|\|$	0.11085	0.00490	0.00280	0.17466	0.20957	0.00280
$\|\|\Psi(P) - P\|\|$	0.170940	0.000000	0.000000	0.000000	0.000000	0.000000
$\|\|\Gamma(v) - v\|\|$	1.189853	0.000000	0.000000	0.000000	0.000000	0.000001
N runs of 100	100	100	100	98	97	100

- MPEC convergence deteriorates
- Equilibrium conditions are satisfied, but estimators start to converge to *wrong* equilibria
(as seen from KL divergence from the data generating equilibrium)

Likelihood correspondence

Lines are costructed using symmetric KL-divergence



Monte Carlo B, run 2: little multiplicity, unstable

Number of equilibria at true parameter: 3
Number of equilibria in the data: 1
Data generating equilibrium: unstable

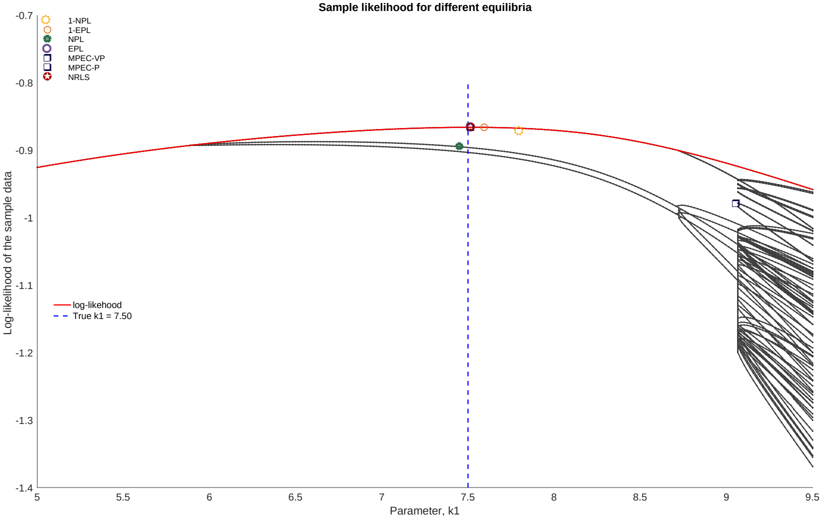
	2step	NPL	EPL	MPEC-VP	MPEC-P	NRLS
True k1=7.5	7.54238	7.39276	7.48044	7.73133	7.63100	7.50176
Bias	0.04238	-0.10724	-0.01956	0.23133	0.13100	0.00176
MCSD	0.17145	0.05608	0.15801	0.72988	0.89874	0.03820
ave log-like	-0.86834	-0.89374	-0.86550	-0.88512	-0.90196	-0.86504
log-likelihood	-21708.60	-22343.58	-21637.54	-22127.91	-22549.06	-21626.12
log-like short	-	-765.242	-11.413	-502.121	-920.643	-0.000
KL divergence	0.02271	0.15996	0.00257	0.11452	0.20182	0.00012
$\|\|P - P_0\|\|$	0.09757	0.20709	0.00619	0.03860	0.02504	0.00307
$\|\|\Psi(P) - P\|\|$	0.160102	0.000000	0.000000	0.000000	0.000000	0.000000
$\|\|\Gamma(v) - v\|\|$	1.126738	0.000000	0.000000	0.000000	0.000000	0.000001
N runs of 100	100	18	100	99	98	100

- NPL estimator fails to converge
- Similar convergence issues for MPEC
- EPL estimator performs well

Aguirregabiria, Marcoux (2021)

Likelihood correspondence

Lines are costructed using symmetric KL-divergence



Monte Carlo B, run 3: discontinuous likelihood

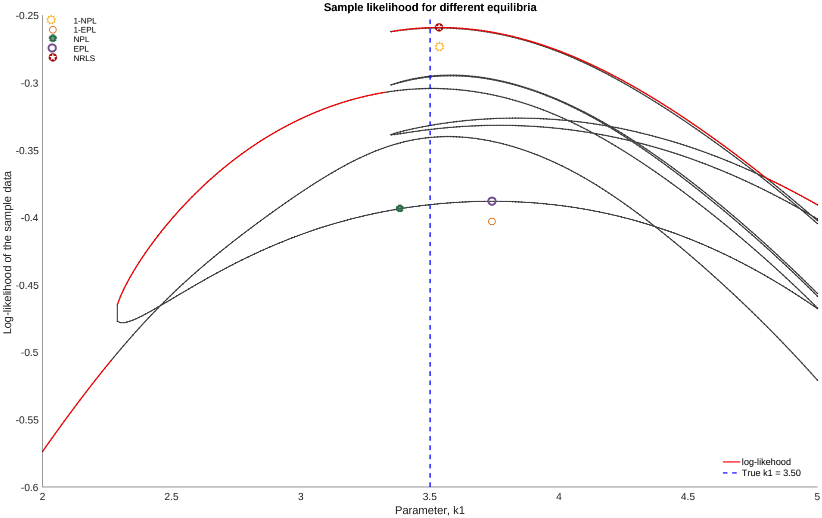
Number of equilibria at true parameter: 9
Number of equilibria in the data: 1
Data generating equilibrium: unstable, near "cliffs"

	2step	NPL	EPL	MPEC-VP	MPEC-P	NRLS
True k1=3.5	3.49739	3.55144	3.64772	3.65943	3.67027	3.50212
Bias	-0.00261	0.05144	0.14772	0.15943	0.17027	0.00212
MCSD	0.13999	0.07133	0.12900	0.12693	0.11583	0.03255
ave log-like	-0.27494	-0.29474	-0.29528	-0.30330	-0.30257	-0.25086
log-likelihood	-1374.721	-1473.695	-1476.425	-1516.503	-1512.847	-1254.320
log-like short	-	-219.375	-222.104	-270.999	-267.523	-0.000
KL divergence	0.01512	0.04889	0.04495	0.04102	0.04078	0.00016
$\|\|P - P_{\mathbf{0}}\|\|$	0.62850	0.86124	0.83062	0.66562	0.65879	0.01610
$\|\|\Psi(P) - P\|\|$	0.763764	0.000000	0.000000	0.000000	0.000000	0.000002
$\|\|\Gamma(v) - v\|\|$	0.852850	0.000000	0.000000	0.000000	0.000000	0.000005
N runs of 100	100	100	100	28	27	100

- Similar convergence issues
- Poor estimates by EPL, NPL and MPEC
  (constraints are satisfied, yet low likelihood and high KL divergence)

Likelihood correspondence

Lines are costructed using symmetric KL-divergence



Monte Carlo B, run 4: massive multiplicity

Number of equilibria at true parameter: 2455
Number of equilibria in the data: 1
Time to enumerate all equilibria (RLS): 10m 39s

	1-NPL	NPL	EPL	NRLS
True k1=3.75	3.70959	3.71272	3.78905	3.74241
Bias	-0.04041	-0.03728	0.03905	-0.00759
MCSD	0.11089	0.06814	0.40716	0.03032
ave log-likelihood	-0.38681557	-0.37348793	-0.45256293	-0.35998461
log-likelihood	-1934.078	-1867.440	-2262.815	-1799.923
log-like shortfall	-	-66.529	-467.607	-0.000
KL divergence	Inf	14.07523	12231.59186	0.32429
$\|\|P - P_0\|\|$	0.82204	0.65580	0.79241	0.07454
$\|\|\Psi(P) - P\|\|$	0.963574	0.000000	0.000000	0.000006
$\|\|\Gamma(v) - v\|\|$	7.020899	0.000000	0.000000	0.000008
N runs of 100	100	18	68	100
CPU time	0.159s	11.262s	4.013s	4.731s

- Severe convergence problems for NPL and EPL
- Poor eqb identification (low likelihood and high KL divergence)
- NRLS has comparable CPU time (much faster than full enumeration)

Monte Carlo C, multiple equilibria in the data

The path forward:

- Assume that the same equilibrium is played in each market over time
- Grouped fixed-effects, groups defined by the equilibria played

1. Joint grouped fixed-effects estimation
    - Estimate the partition of the markets into groups playing different equilibria together with θ
    - For each market compute maximum likelihood over all equilibria and "assign" it to the relevant group (estimation+classification)
    - Computationally very demanding: BnB market-by-market, non-parametric refinement has no bite
2. Two-step grouped fixed-effects estimation
    - Step 1: partition the markets based on some observable characteristics (K-means clustering)
    - Step 2: estimate θ allowing different equilibria in different groups
    - Small additional computational cost!

Bonhomme, Manresa (2015); Bonhomme, Lamadon, Manresa (2022)

NRLS estimator for directional dynamic games

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

NRLS nested structure:

1. Each stage game $\rightarrow$ non-linear solver, specific to the model
2. Combining stage game solutions to full game MPEs $\rightarrow$ State Recursion algorithm
3. Solving for all MPE equilibria $\rightarrow$ Recursive Lexicographic Search
4. Structural estimation $\rightarrow$ 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 $\rightarrow$ less computation
- Computationally trackable, better performance with more data
- Fully robust to multiplicity of equilibria