Nash Memory Mechanism for Coevolution

Sevan G. Ficici

Outline

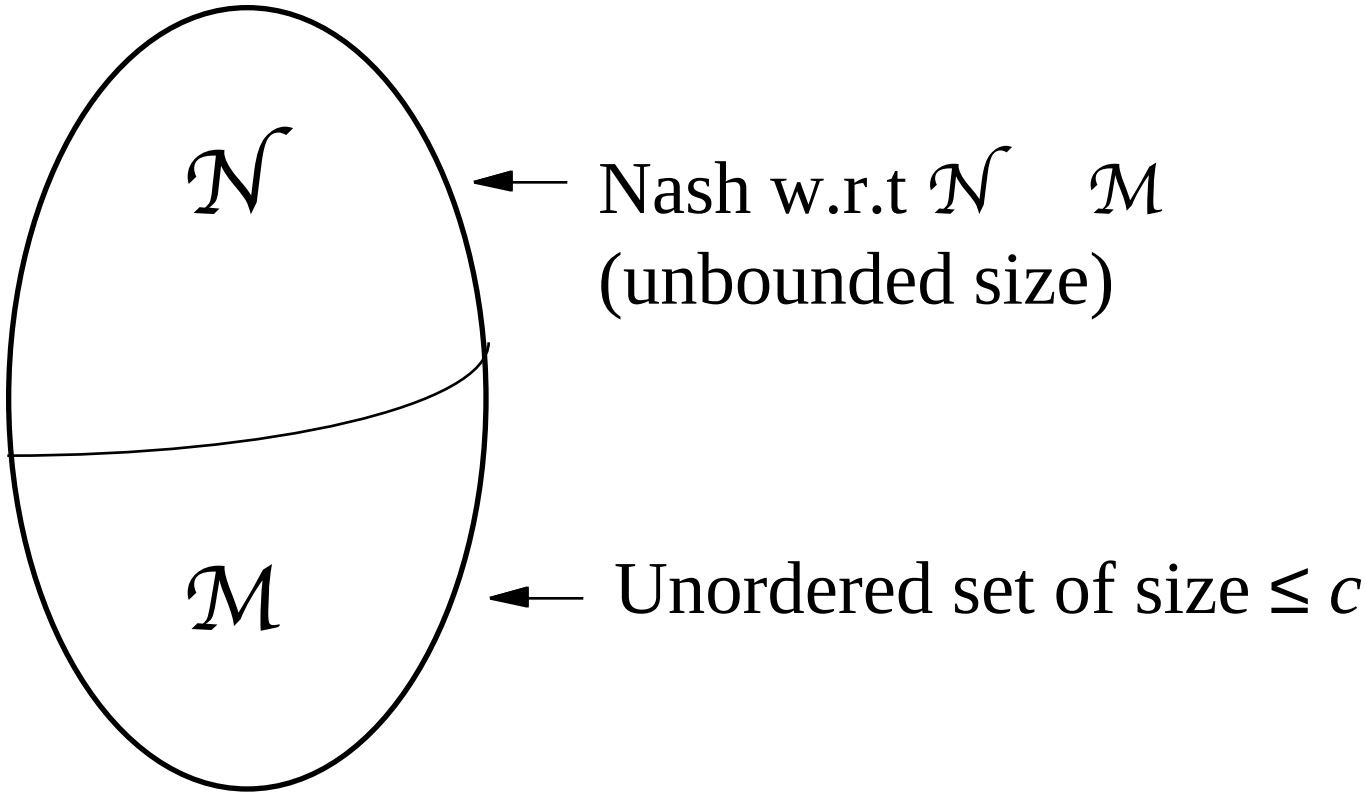
- Game theory and memory mechanism
  - Definition and operation
  - Models of mechanism
  - Experiments using mechanism
- MOO and game theory

Memory Mechanisms

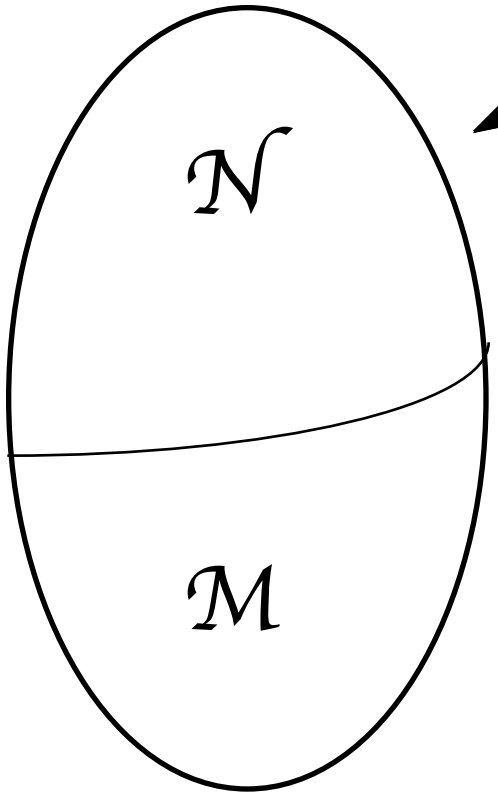
- Intent is to save “good” strategies for subsequent testing (why not for search?)
- Examples
  - Hall of fame, dominance tournament
- We show a mechanism where “good” has a formal game-theoretic meaning
- For zero-sum, symmetric games

Simple Memory Mechanism

- Two mutually exclusive sets of strategies $\mathcal{N}$ and $\mathcal{M}$



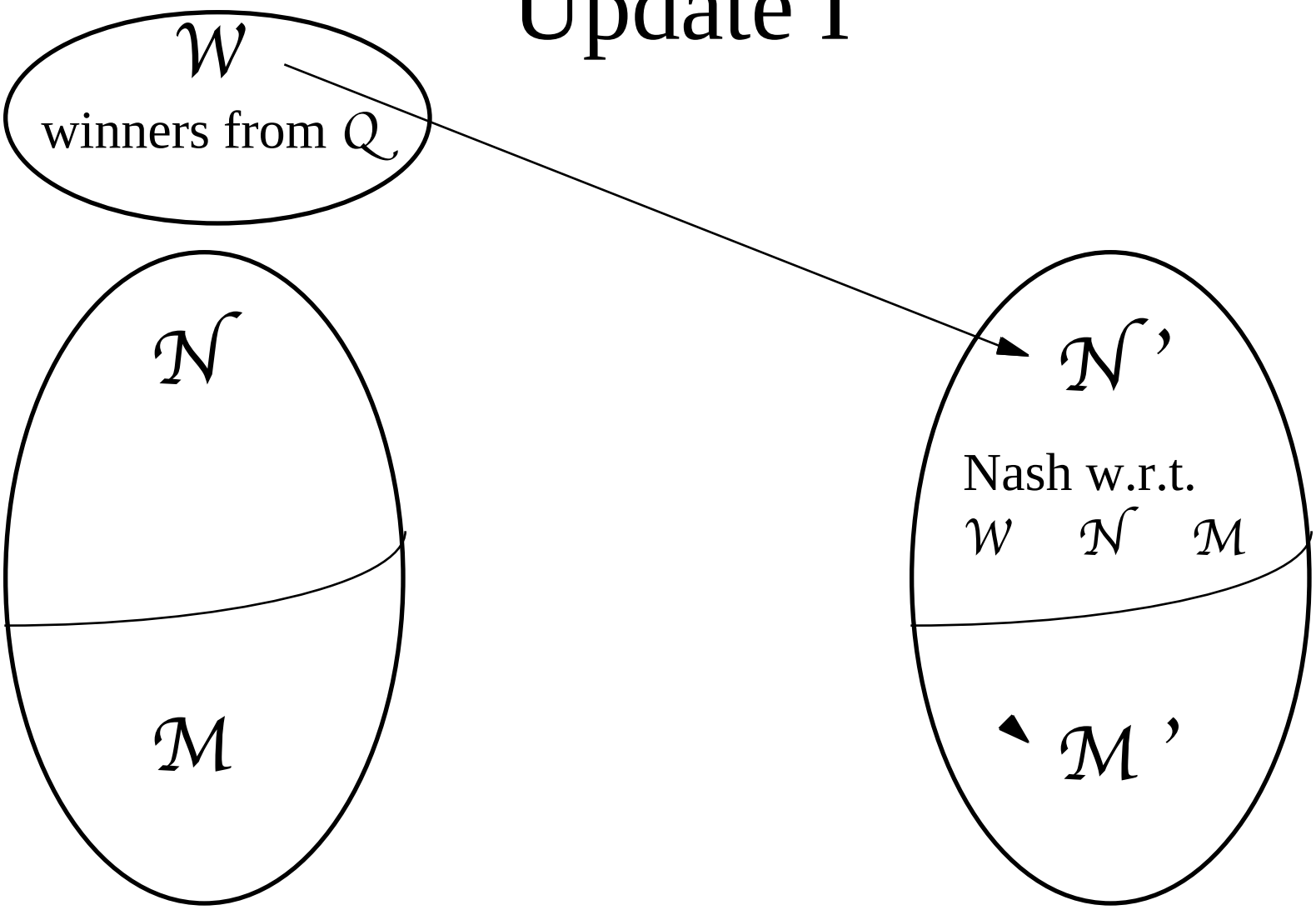
Testing



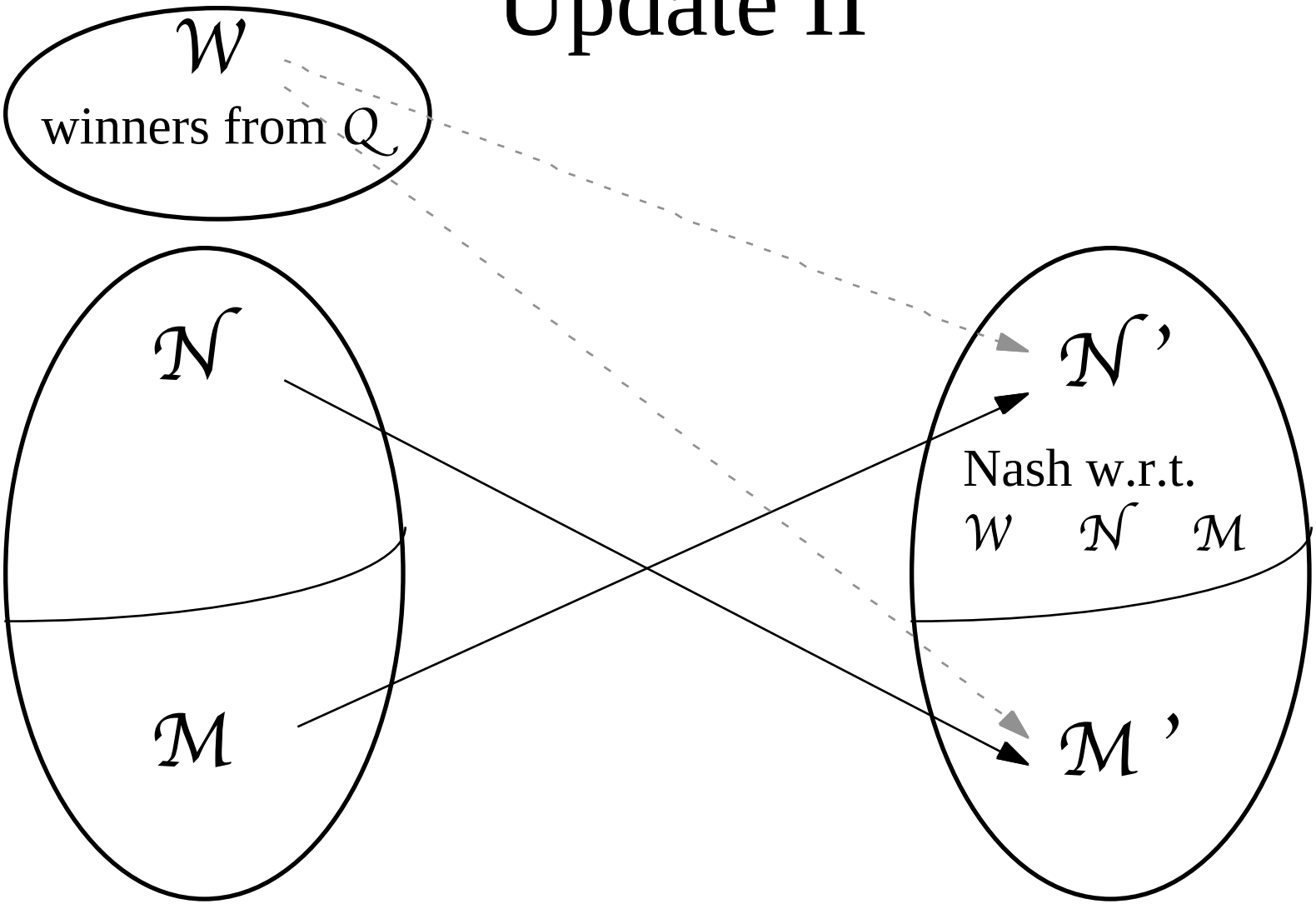
$\mathcal{Q}$ (set of strats from search)

- for each q $\mathcal{Q}$:
  - if $E(q, \mathcal{N}) \leq 0$, then discard
  - else use q for update

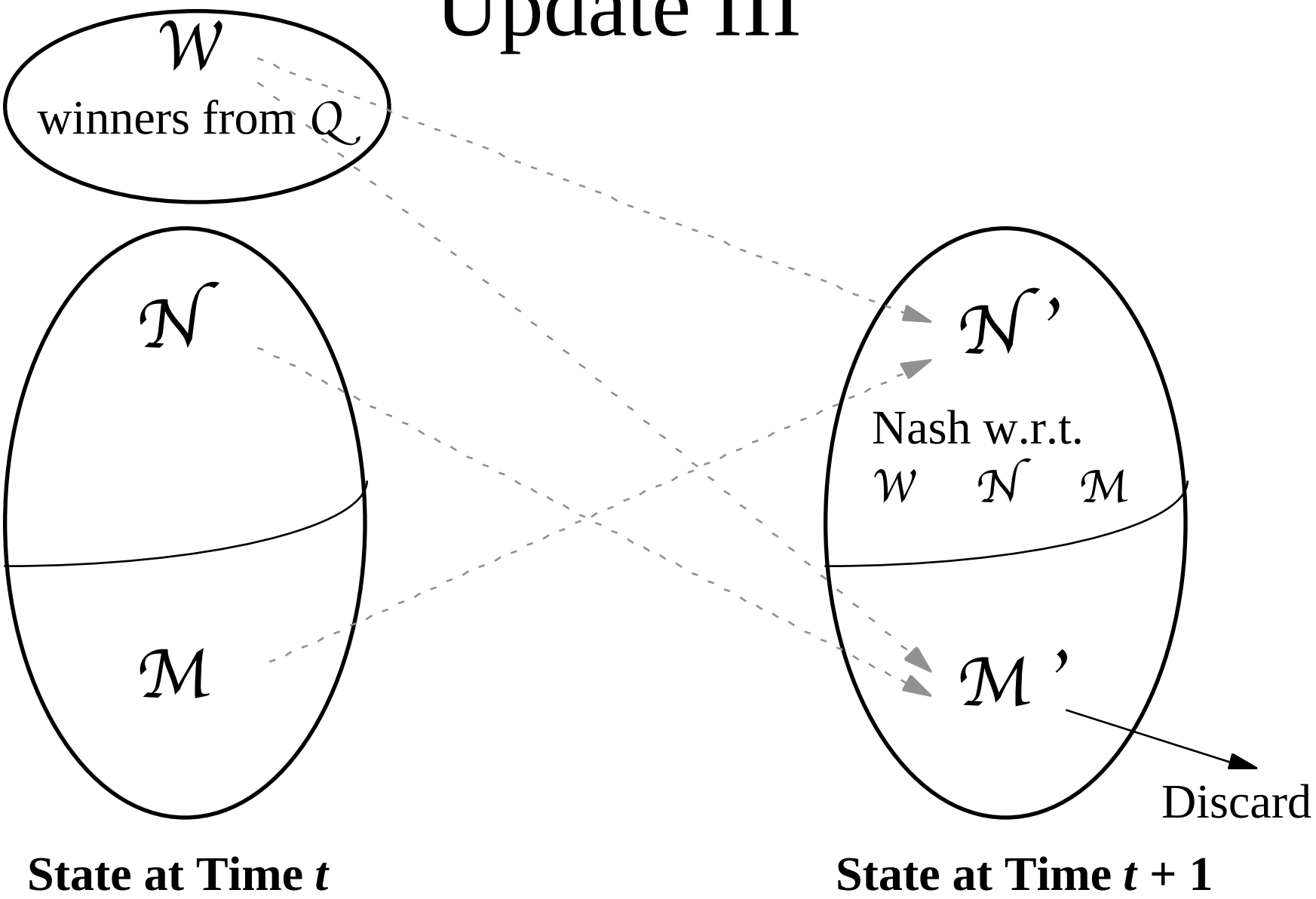
Update I


$\mathcal{W}$
winners from Q
$\mathcal{N}$
$\mathcal{M}$
$\mathcal{N}'$
Nash w.r.t.
$\mathcal{W}$ $\mathcal{N}$ $\mathcal{M}$
$\mathcal{M}'$
State at Time *t*
State at Time *t* + 1

Update II


$\mathcal{W}$
winners from Q
$\mathcal{N}$
$\mathcal{M}$
$\mathcal{N}'$
Nash w.r.t.
$\mathcal{W}$ $\mathcal{N}$ $\mathcal{M}$
$\mathcal{M}'$
State at Time *t*
State at Time *t* + 1

Update III


$\mathcal{W}$
winners from Q
$\mathcal{N}$
$\mathcal{M}$
$\mathcal{N}'$
Nash w.r.t.
$\mathcal{W}$ $\mathcal{N}$ $\mathcal{M}$
$\mathcal{M}'$
Discard
State at Time *t*
State at Time *t* + 1

Intent of Mechanism

- There exists a universe of strategies S
- We search S with some heuristic $\mathcal{H}$
- Have $\mathcal{N}$ become a better and better approximation of Nash of S over time

Challenges

- Not necessarily looking for a singleton
- More like a combinatorics problem

- Guaranteed if capacity of $\mathcal{M}$ is infinite, but how much work?
- What happens if capacity is finite?

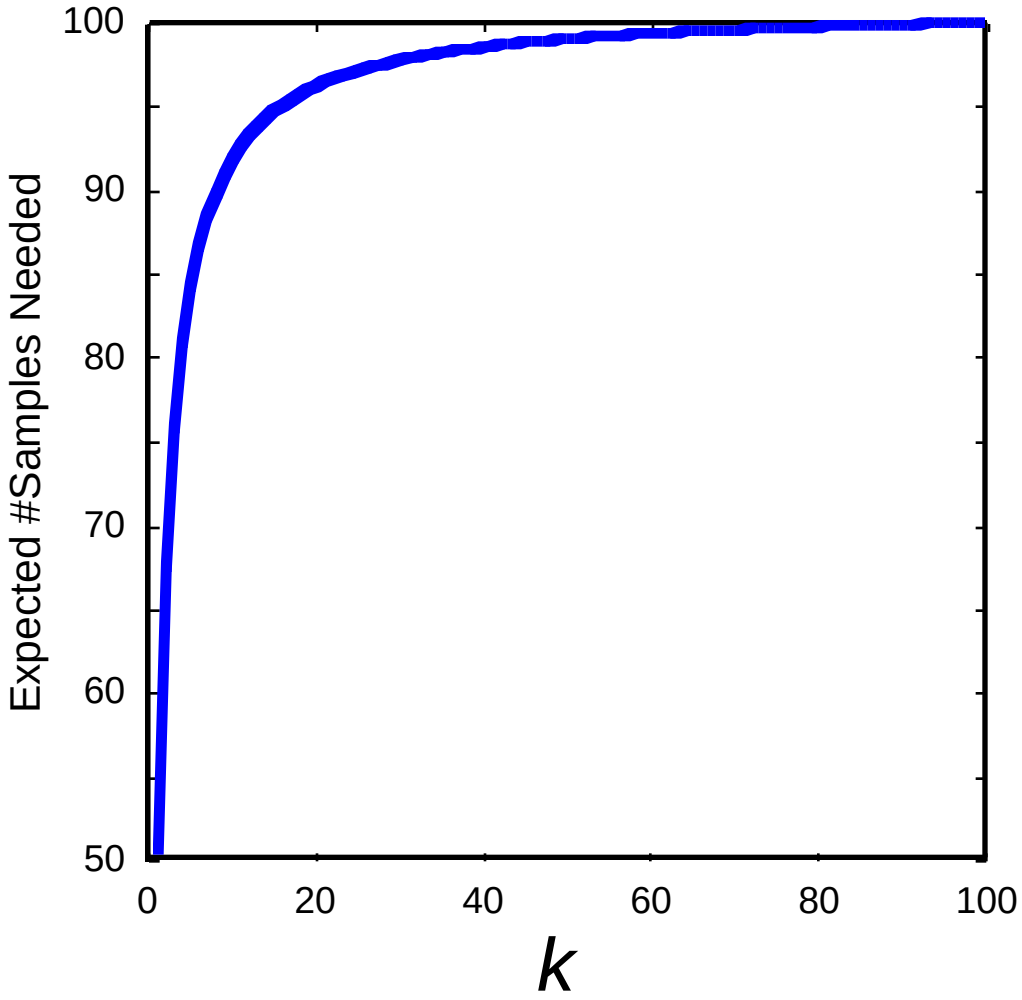
Model I

- Infinite memory
- Heuristic is random search

- Container of n items, of which k are desired
- We sample without replacement from container (one item at a time)
- How many samples needed to collect all k desired items, on average?

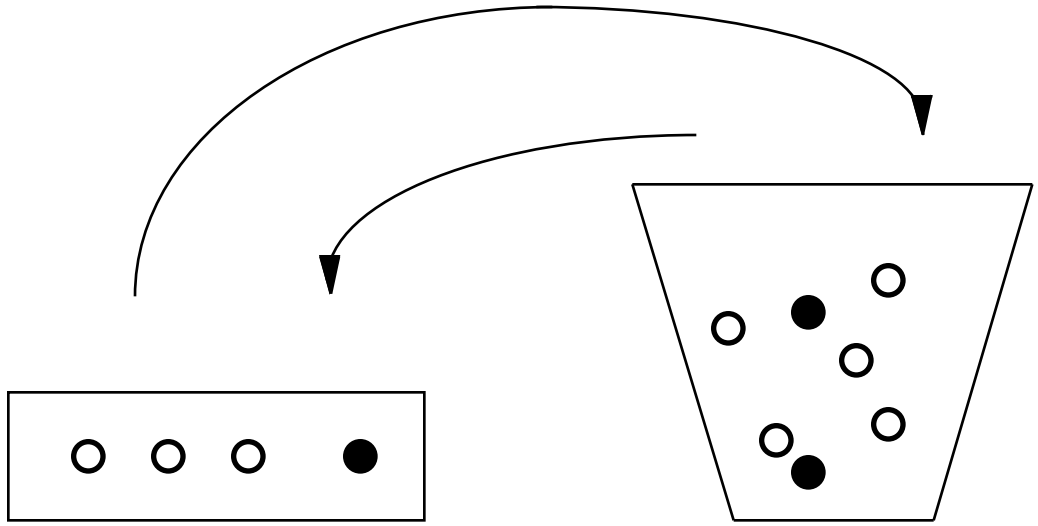
Results

$n = 100$



Model II

- Sample with *delayed* replacement, random search heuristic
- Accumulator has finite capacity for non-desired items
- Accumulator only knows number of undesired, not identity
- Pick at random from accumulator for replacement



Accumulator with capacity c, $c < n - k$, for non-desired items

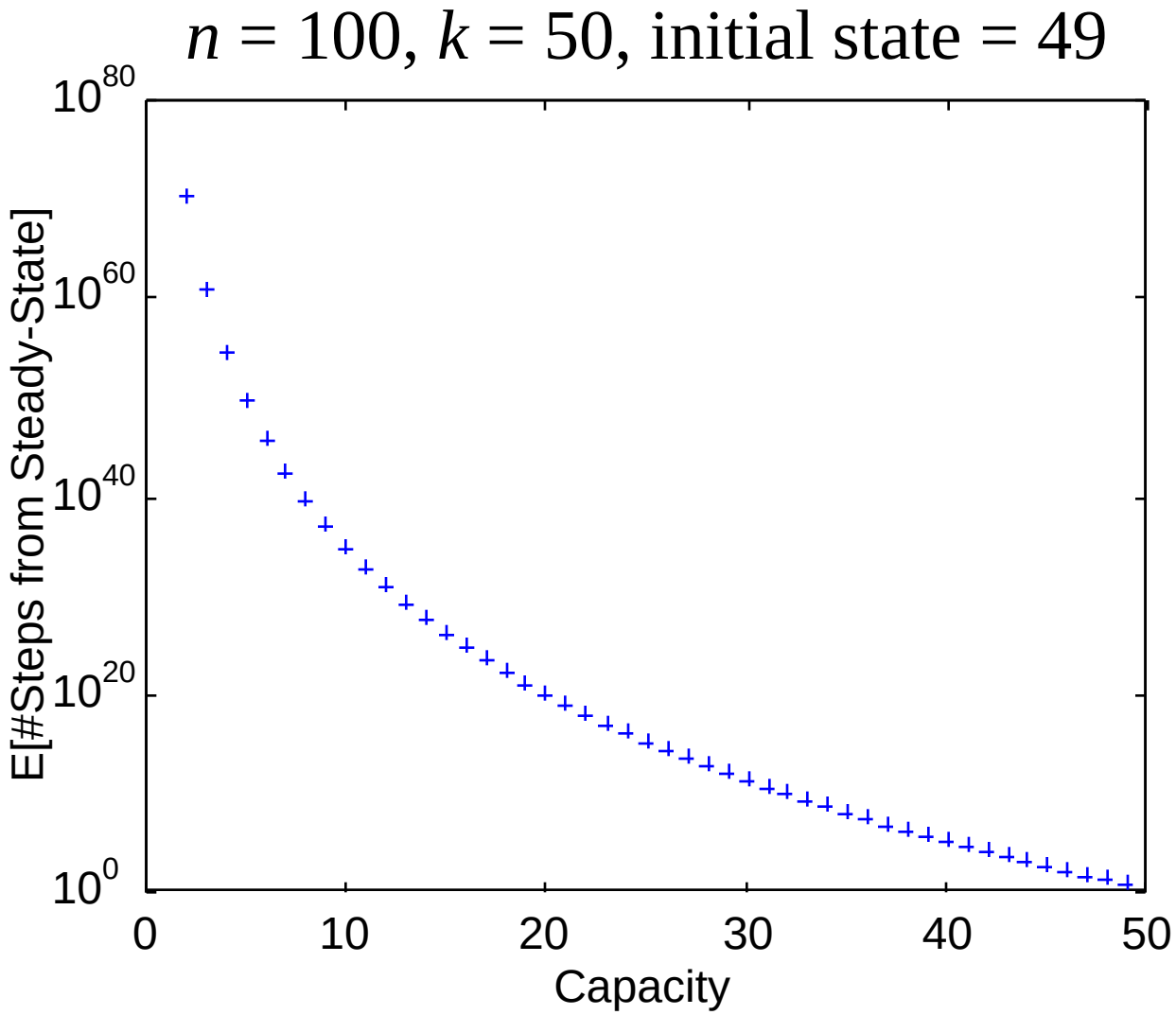
Set of n items, k desired

Model II

- Model with Markov chain
- Accumulator at capacity
- Chain states represent no. desired items
- At steady-state, where $n = 100$, $k = 50$:

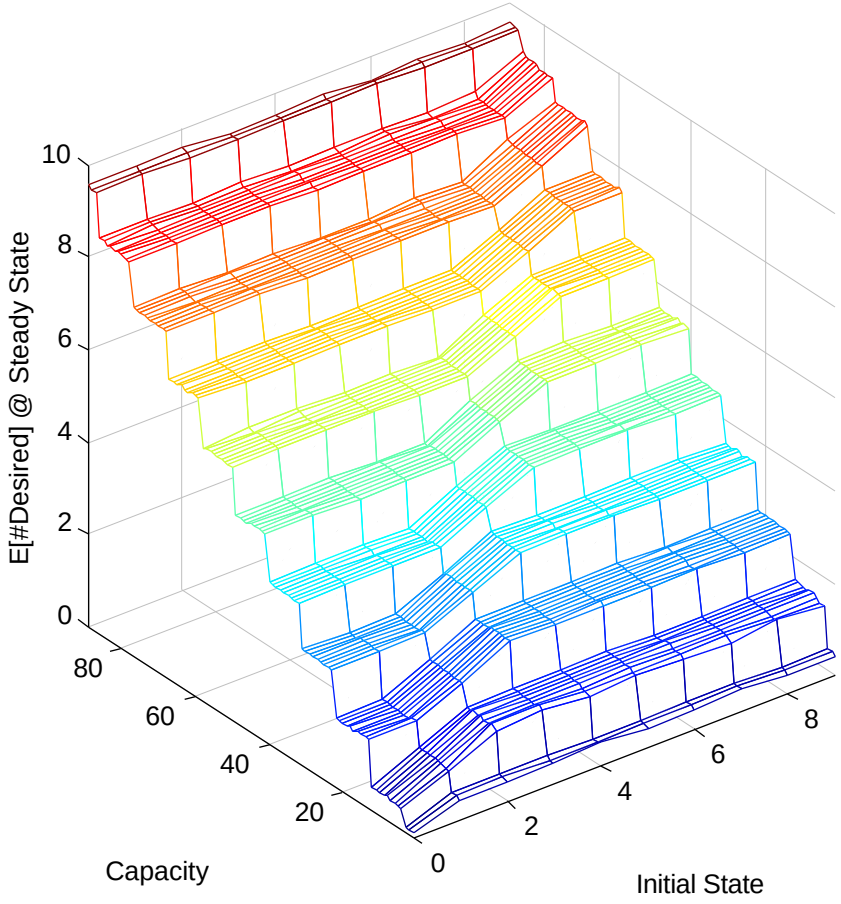
  #desired = #undesired, that is c
- Update costs:
  - expected no. of updates to reach steady state
  - expected no. of updates to reach chain state where all k items are in accumulator (from steady state)
  - (ignoring no. of updates to fill accumulator)

#Updates from Steady-State



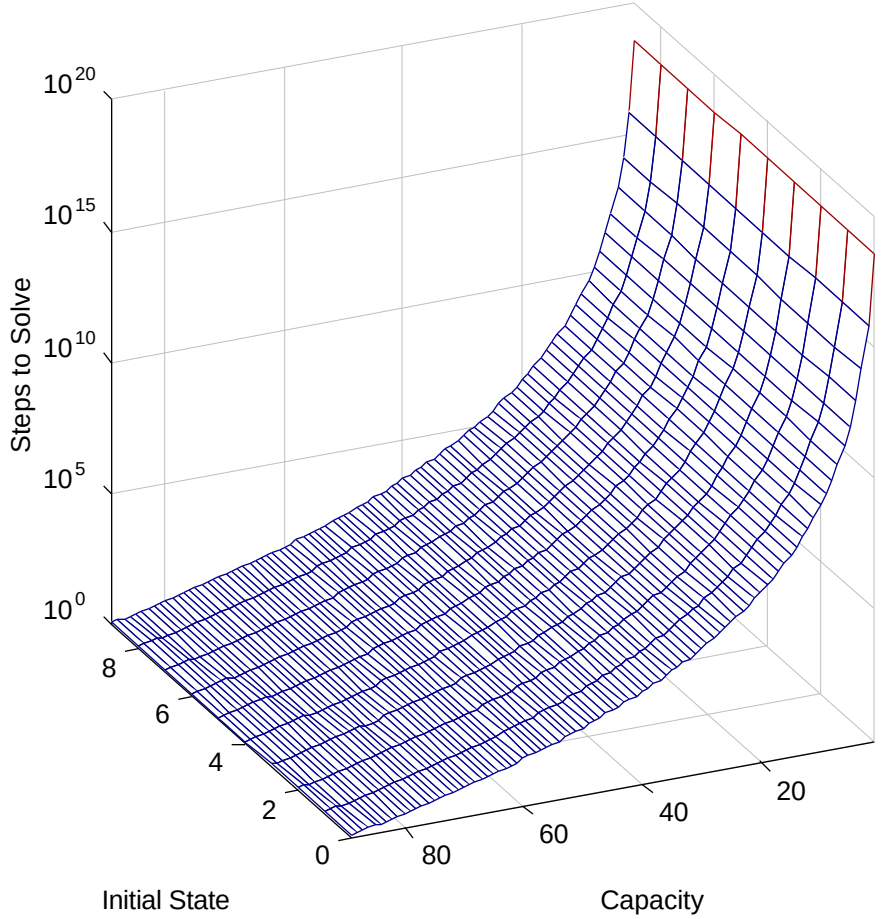
Steady State

$n = 100, k = 10$



#Updated from Steady State

$n = 100, k = 10$



Models Review

- Assuming random search and infinite capacity: Need to search most of space
- Assuming random search, finite capacity, random selection for replacement: May need astronomical amount of work
- Search for *k* desired items can also be a framework for judging performance of algorithms with other solution concepts

Models Review

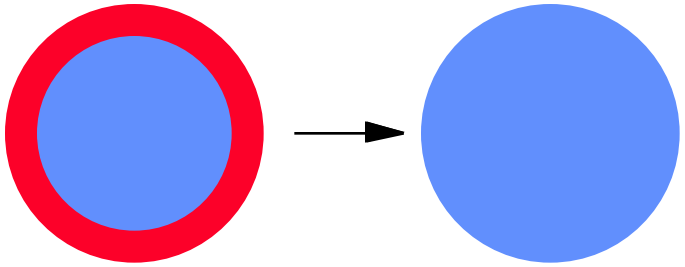
- Need to refine Model II to get better estimates of no. steps to obtain all *k* desired
- Collect data from the memory mechanism and compare (in progress)

Experiment I

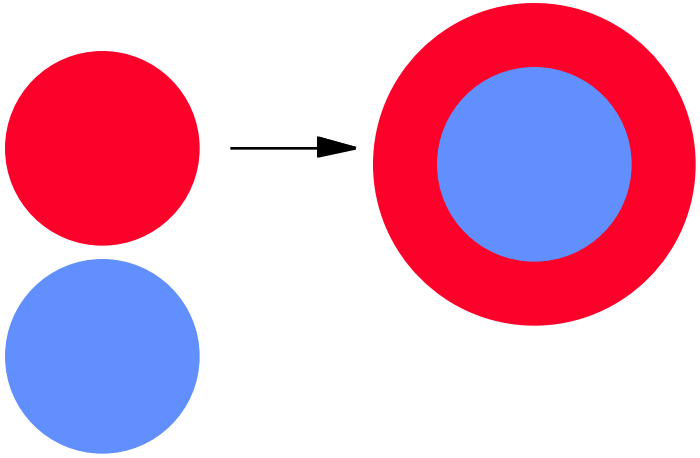
- Tests against $\mathcal{N}$ return reals in range [-1, 1]
- Use tests against $\mathcal{N}$ as an adaptive, quasi-static evaluation function
  - Evolve population against $\mathcal{N}$ (30 generations)
  - Update $\mathcal{N}$ with best of evolved individuals
  - Restart evolution from similar initial condition
  - Iterate until $\mathcal{N}$ is no longer learnable -- no avenue for further improvement vs. $\mathcal{N}$

Comment

- Inversion of usual arrangement



Job of learner is to span gap between its current abilities and demands placed upon it by teacher

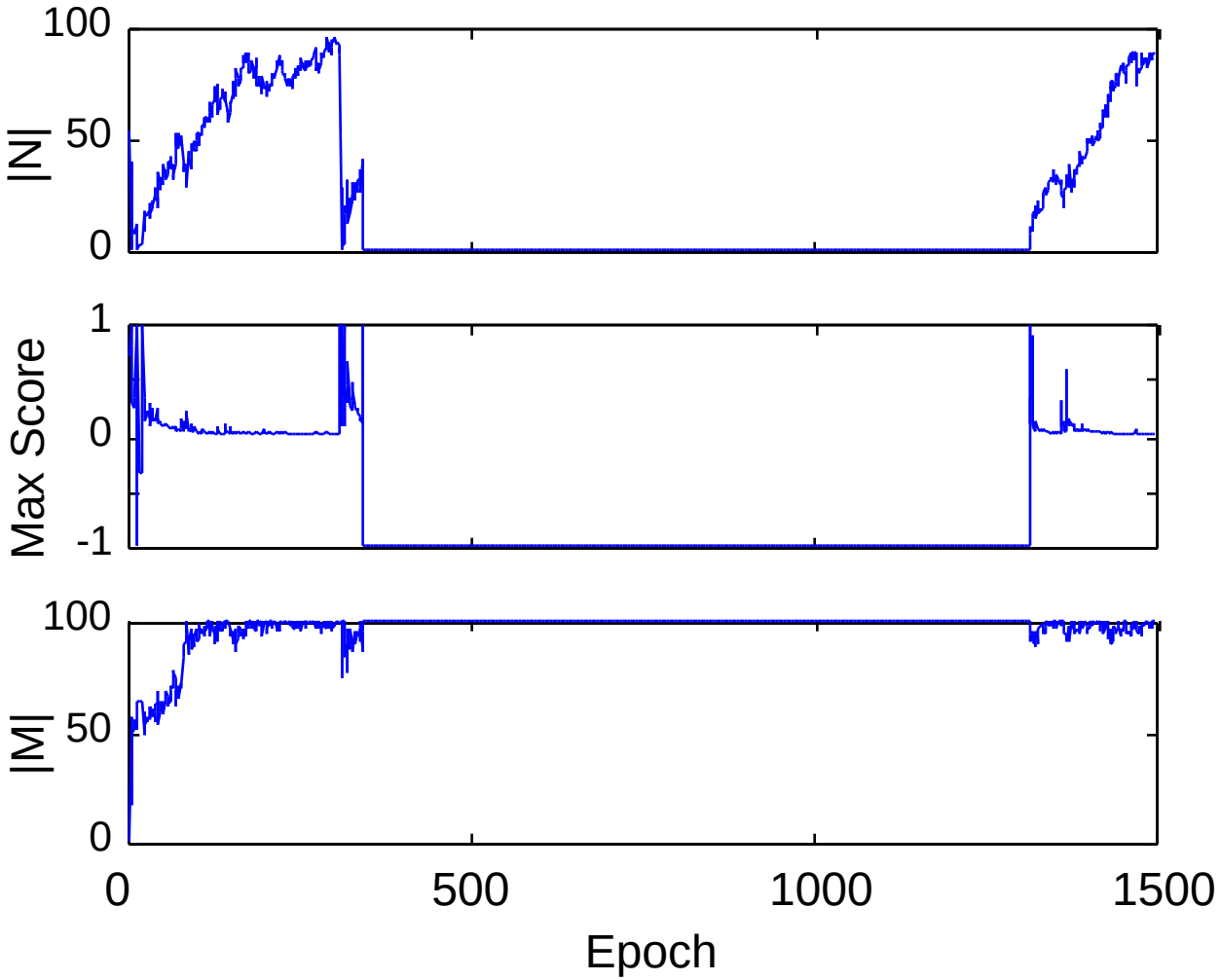


Job of teacher is to become unlearnable by learner. In so doing, the teacher has induced knowledge.

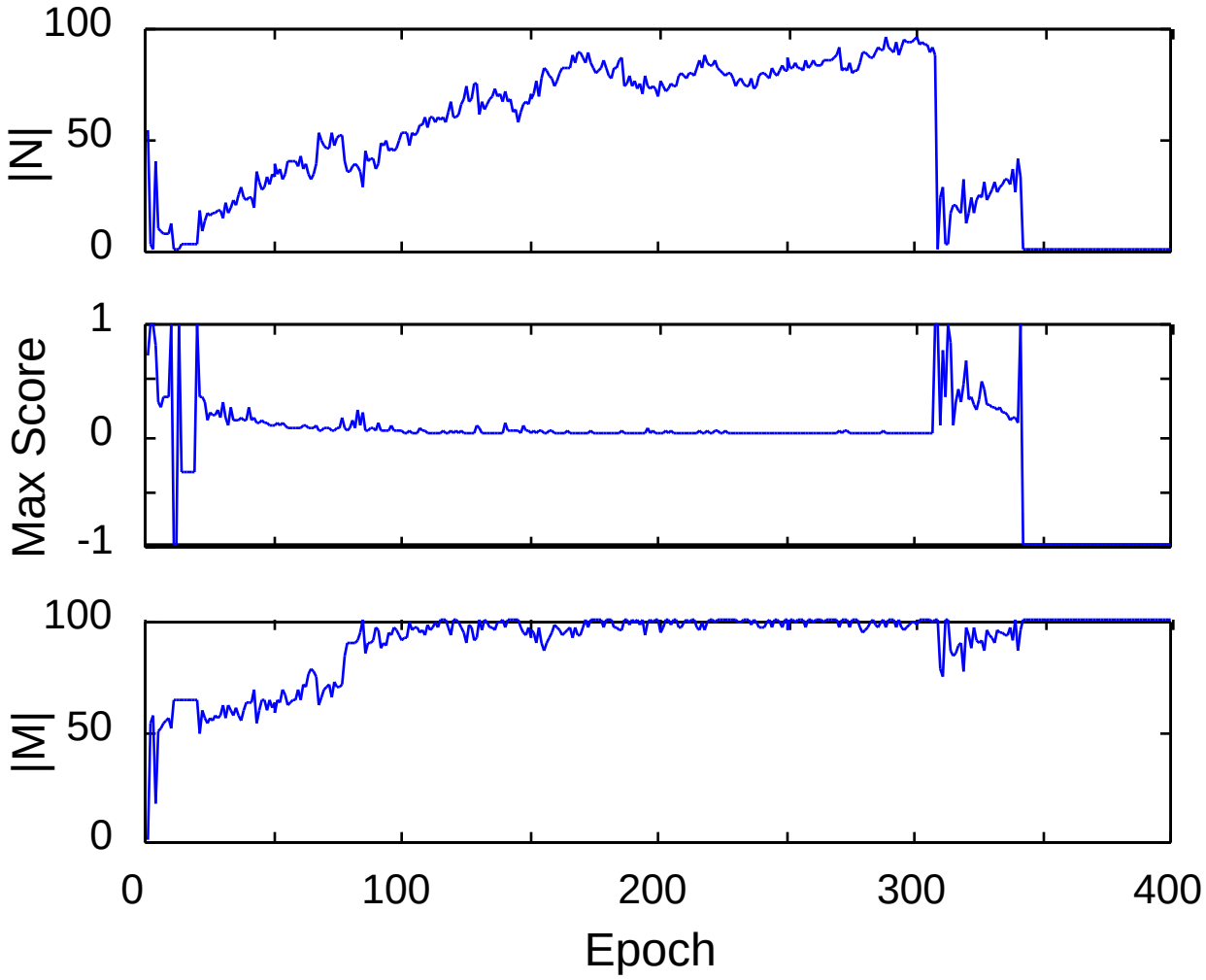
Another Comment

- There is co-adaptation between memory and evolving population
- But, the memory is the learner (using a non-evolutionary learning mechanism)
- And the teacher is the EA

Sample Run



Zoom



Experiment II

- Once $\mathcal{N}$ is unlearnable
  - Save its contents and use it as a ratchet
  - That is, use it to initialize restarts as a heurisitic to endow evolving population with new skills
- Iterate this process and see how things progress

Show Matlab

Utility of Solution Concept

- How well is the algorithm doing?
- How can we tell?
- Need a solution concept

Numbers Game

(finite game in quadrant I)

- Non-domination
  - All but four strategies are non-dominated
- Nash equilibrium
  - Top right corner is the only Nash strategy
- “Go up and to the right,” “Higher values are better”
  - These are not solution concepts
  - Avoid! Optimality is external to the game

Pareto HC: Concept?

- Domination used in algorithm
- But, what solution concept is implemented?
- It goes up and to the right, but is that correct?
- What would it do if all teachers were available?
- Initial conditions probably matter

Game Theory and MOO

- MOO gives tradeoff surface
- Typically pick a single point on surface for building
- If we can build a “polymorphic” entity, then it should behave like a probability distribution over tradeoff surface
- That’s what Nash does for games