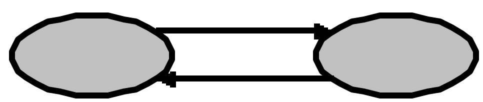
Formal Analysis of Hierarchical State Machines

Rajeev Alur

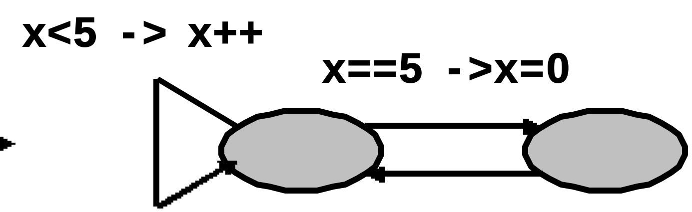
University of Pennsylvania

In honor of Zohar Manna
Taormina, June 2003

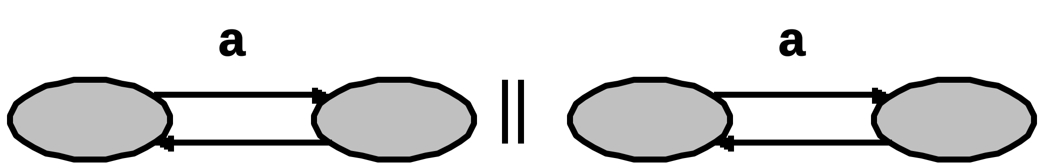
State-Machine Based Modeling



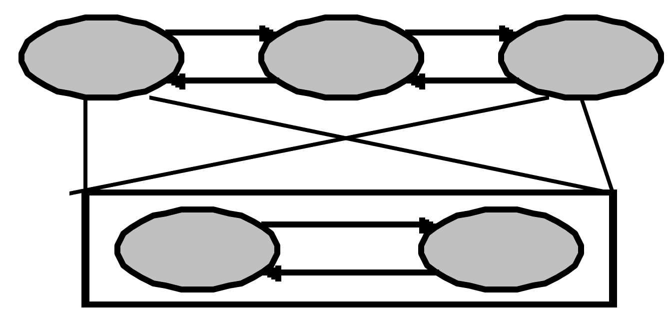
Finite State Machines



Extended FSMs



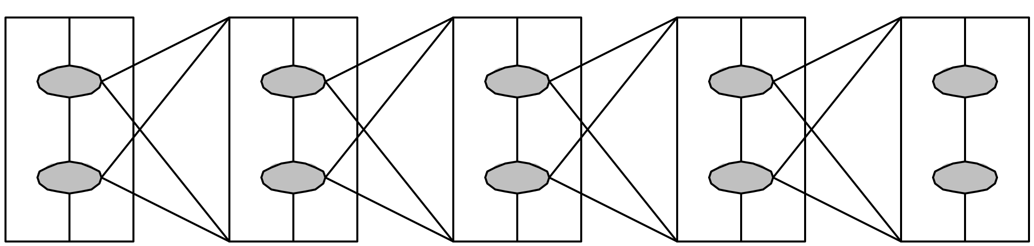
Concurrent FSMs



Hierarchical FSMs

Hierarchy -> Succinctness

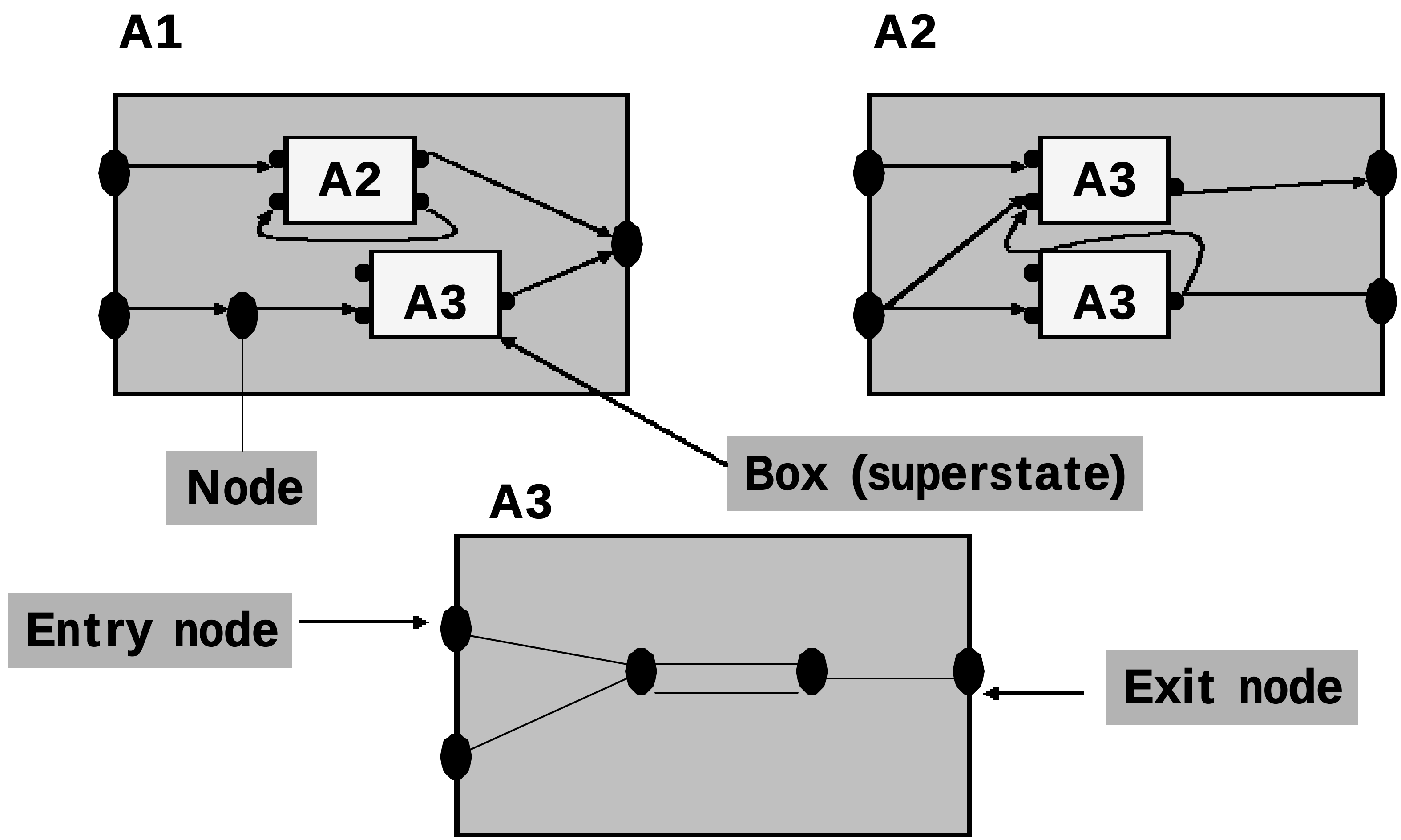
- Concurrent FSMs are exponentially more succinct than FSMs
- Extended FSMs (boolean variables) are also exponentially more succinct
- Hierarchical FSMs are also exponentially more succinct than FSMs due to sharing
- Intuition: can count succinctly: e.g. can express a^n with log n levels of nesting



Motivation

- Concurrent FSMs and Extended FSMs well understood and supported by model checkers
- Hierarchy common in modern software design languages (e.g. Statecharts, UML)
- Goal 1: Theoretical foundations for hierarchical state machines (succinctness, complexity, formal semantics, ….)
- Goal 2: What’s the best way to analyze Hierarchical FSMs ? (avoid flattening, exploit hierarchy/sharing)

Hierarchical State Machine

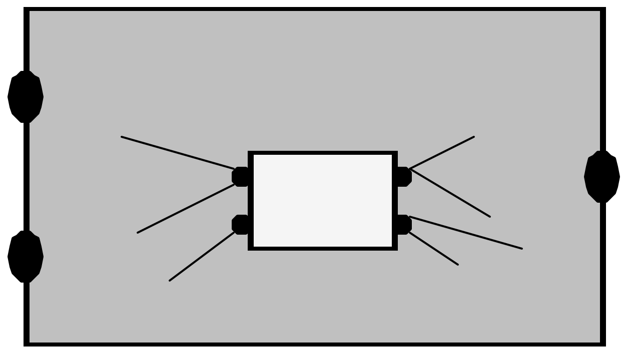


Reachability

- Underlying transition system (expansion)
  - State records context (seq of boxes) and node
  - Transitions: internal, calls, returns
  - Size: exponential in nesting depth (bound is tight)
- Concurrent FSMs are exponentially more expensive than FSMs (PSPACE complete)
- Extended FSMs (boolean variables) are also exponentially more expensive (PSPACE complete)
- Reachability for Hierarchical FSMs is in P
- Intuition: Every nested FSM needs to be searched just once for each entry point

Reachability

- On-the-fly enumerative search algorithm tabulates the results of searching a component



- Complexity bound: PTIME complete
- $O(n\ k^2)$ algorithm where n is total size, and k = $\max_i$ min (entry, exit nodes of component Ai)

Talk Outline

✓ Motivation

➲ Automata and Succinctness

❑ Temporal Logic Model Checking

❑ Modeling Language and Tool

Hierarchical Automata

- Hierarchical state machines with edges labeled by alphabet symbols, and initial/final nodes can be viewed as language generators
- $\{w \# w^R \mid |w| = n\}$ has O(n) generator
- Language emptiness: easy (same as reachability)
- Emptiness of intersection of 2 automata is Pspace-complete
- Universality and language equivalence are Expspace-complete
  - Upper bound: Expansion gives an exponential-sized nondeterministic automaton
  - Lower bound: Can guess the error in the encoding of computation of expspace Turing machine, and count succinctly
  - Recall: for pushdown automata, emptiness is in P, but emptiness of intersection and universality are undecidable

Concurrent Hierarchical Automata

- Concurrency (synchronization on common symbols) and hierarchy nested. A component is
  - parallel composition of already defined components, or
  - Hierarchical state machine with nodes and boxes, with boxes mapped to already defined components
- If each hierarchical component has k nodes/boxes, a parallel component has at most d components, and nesting depth is m, then expansion has size $O(k^{d^m})$
- Reachability is expspace-complete
- Universality is 2expspace-complete

Reachability Summary

What is the cost of concurrency and hierarchy ?

FSM : NLogSpace

Concurrent : PSPACE

Hierarchical: PTIME

Concurrent Hierarchical: EXPSPACE

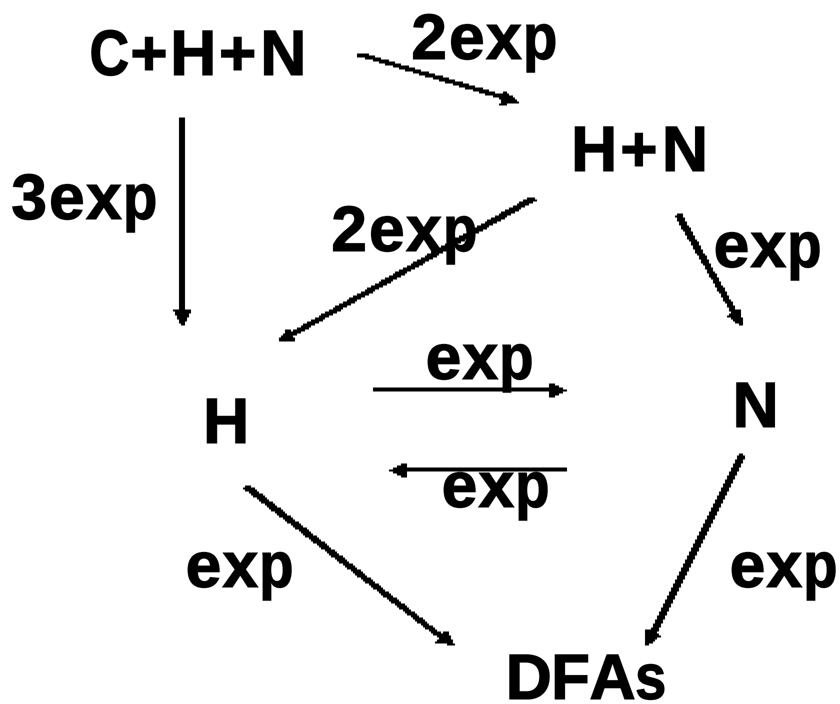
Succinctness

- Standard automata: NFA are exp succinct than DFA (consider {w | exists i. $w_i=w_{n+i}$ })
- NFA are exp more succinct than DHA (det hierarchical) for same reason
- DHA exp more succinct than NFA (consider {$w\#w^R$ | $|w|=n$})
- NHA (nondet hierarchical) are doubly-exp more succinct than DHA/DFA (consider { w | exists i. $w_i = w_{i+2^n}$ })
- Concurrent hierarchical automata are doubly-exp succinct than NHA/NFA and triply-more succinct than DFA/DHA (consider {$w^0\#w^1\#\ldots$ | exists i. $w^i=w^j$ and $|w^i|=2^n$})

Succinctness Summary

Features:

- N: Nondeterminism
- H: Hierarchy
- C: Concurrency



Talk Outline

✓ Motivation

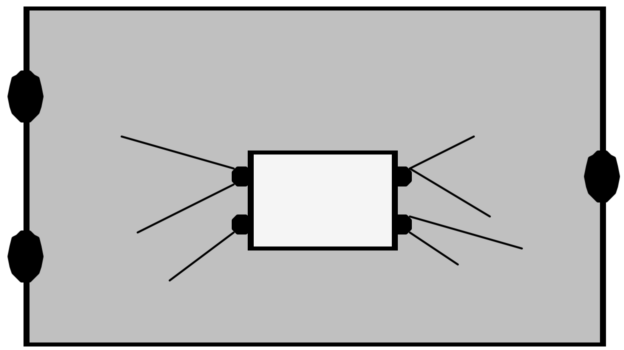
✓ Automata and Succinctness

➲ Temporal Logic Model Checking

❑ Modeling Language and Tool

Cycle Detection

- Given a set T of nodes, is there a cycle containing a node in T and reachable from initial nodes?



- Relevant information about a box: for entry e and exit x, is an accepting cycle reachable from e, is x reachable from e along a path containing a node in T, is x reachable from e
- Complexity same as reachability (Ptime-complete, and in time $O(nk^2)$)

LTL Model Checking

- Given a hierarchical structure K (HSM with nodes labeled with atomic propositions P), and Buchi automaton A over 2^P, to check if some execution of a is accepted by A
  - Take product of K with A, and solve cycle detection
  - Complexity $O(a^2k^2|A|\ |K|)$, where A has a states

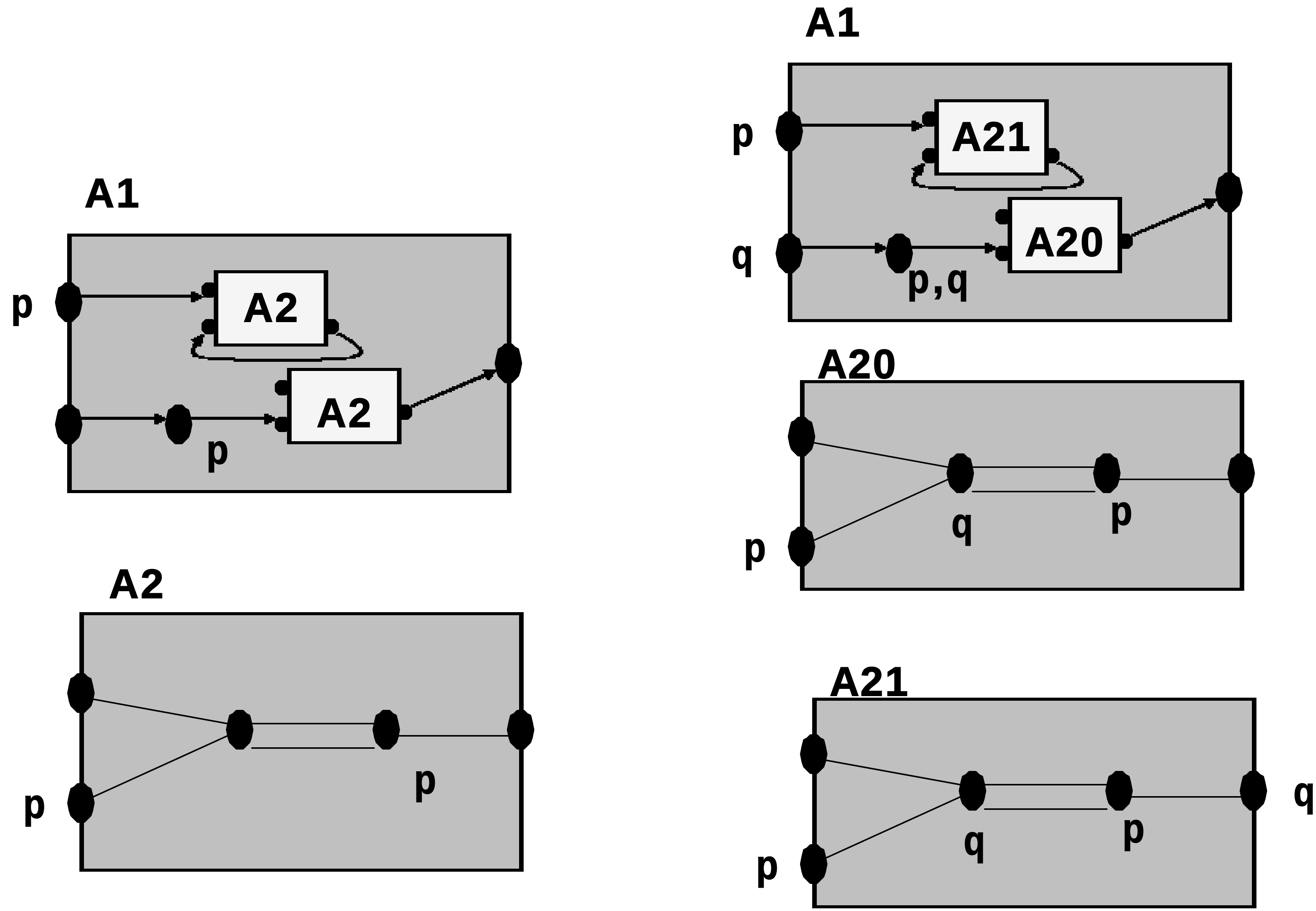
- To check if all infinite executions of K satisfy LTL formula f over P, construct Buchi automaton $A_{\sim f}$, take product, and solve cycle detection
  - Complexity $O(k^2\ |K|\ 8^{|f|})$

Branching Time Logics

- Given a Hierarchical structure K, and CTL formula f, label nodes of K with subformulas of f (process in increasing order of complexity as usual)
  - A node u of component Ai is labeled with f’ if u satisfies f’ in all contexts Ai appears in

- Processing a formula may require splitting

Sample case: Processing q=EX p



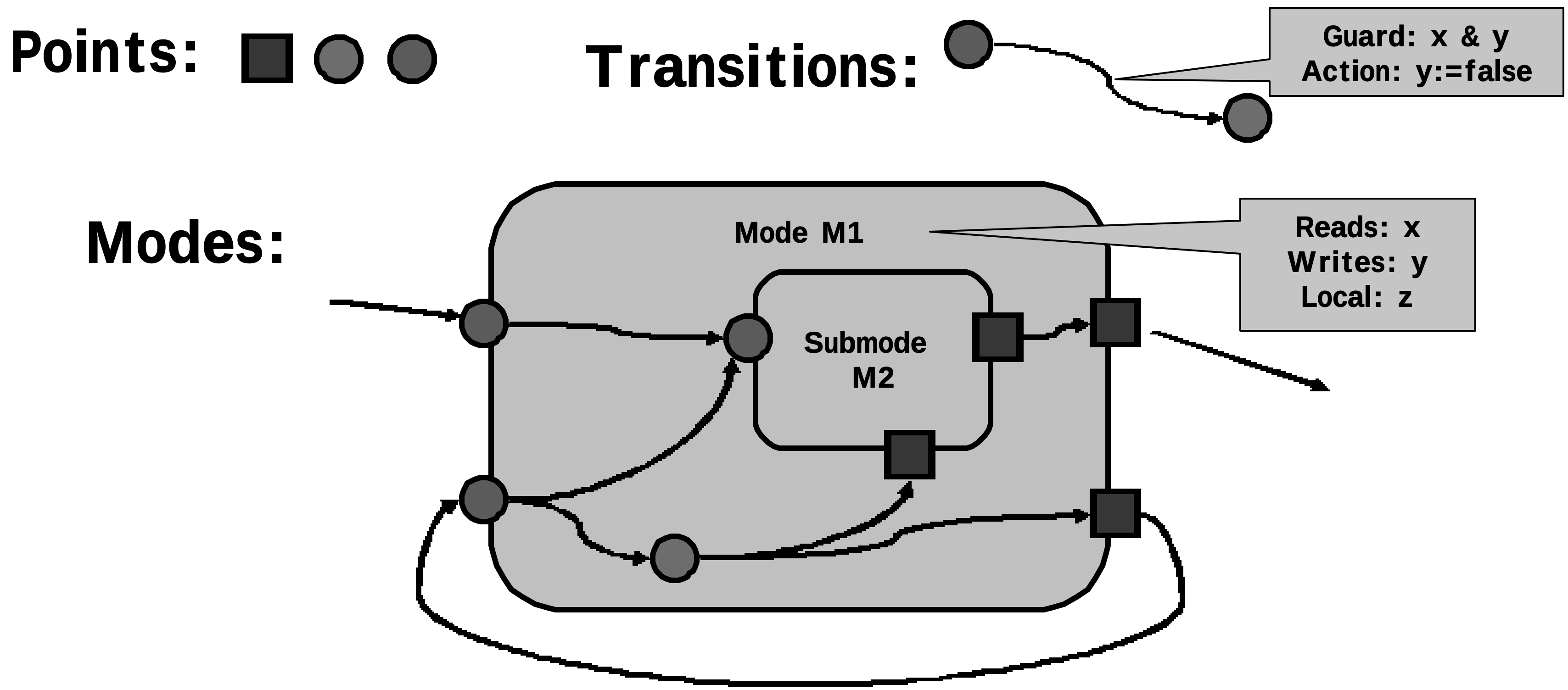
CTL Model Checking

- Handling of Until and Always formulas more subtle
- If every component has at most d exits and k entries, then time complexity is $O(k^2\ |K|\ 2^{|f|d})$
- PSPACE complete problem
- Pspace hardness in both parameters: size of formula f and number of exits d

Talk Outline

- ✓ Motivation
- ✓ Automata and Succinctness
- ✓ Temporal Logic Model Checking
- ➲ Hermes: Modeling Language and Tool

Hierarchical Modules

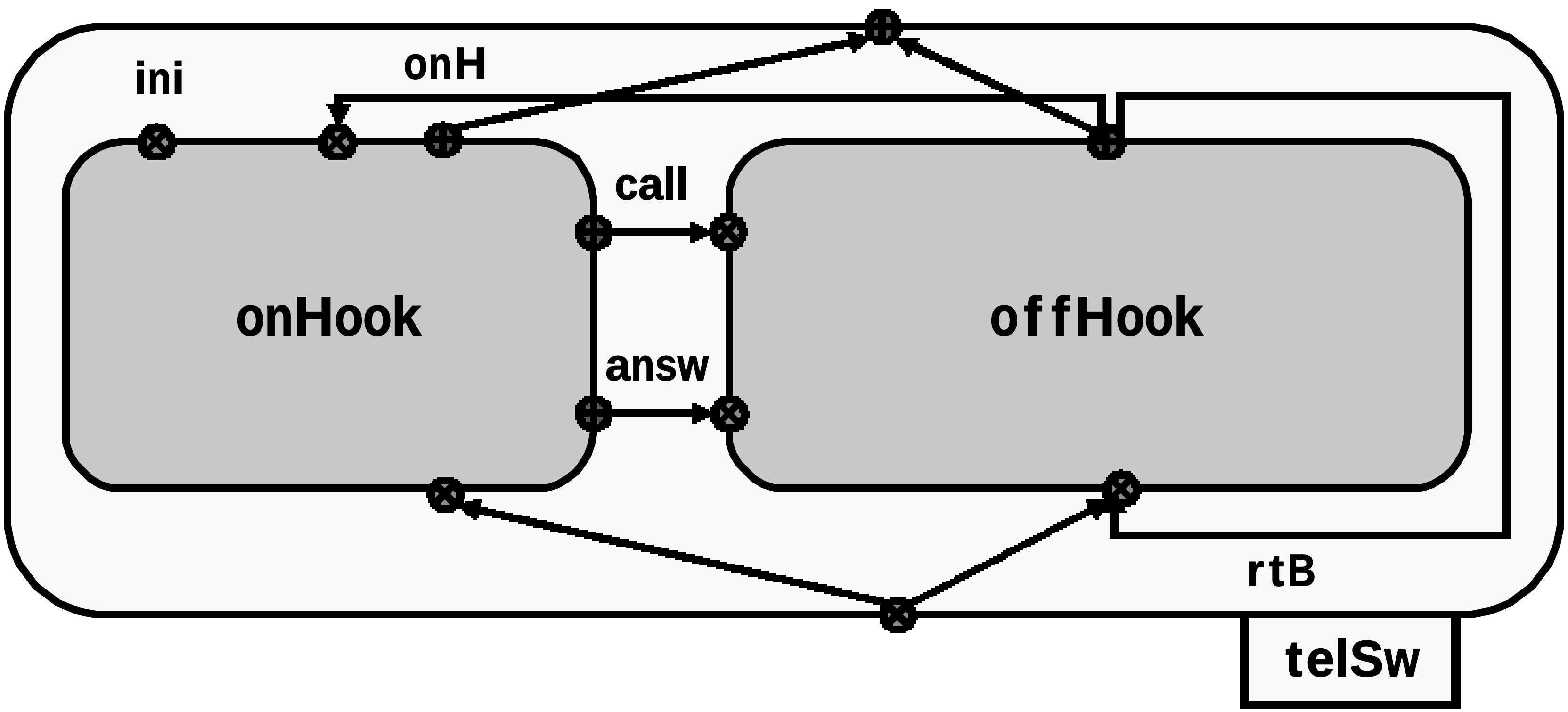


Concurrent, Extended, Hierarchical FSMs
Well-defined interface: Entry/exit points, Read/write variables
Formal, compositional trace-based semantics with refinement calculus

From Statecharts to Modes

Obstacles in achieving modularity

- Regular transitions connect deep nested modes. -> Entry/exit points (control interface)
- Group transitions implicitly connect deep nested modes -> Default points (control interface)
- Nested state references break encapsulation. State references -> Scoping of variables (data interface)



Semantics of Modes

Game Semantics

- Environment round: from exit points to entry points.
- Mode round: from entry points to exit points.

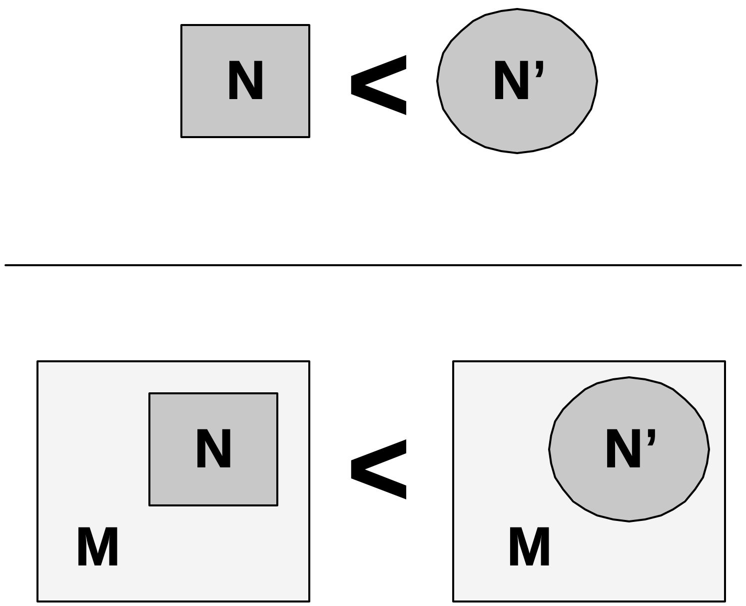
The set of traces of a mode

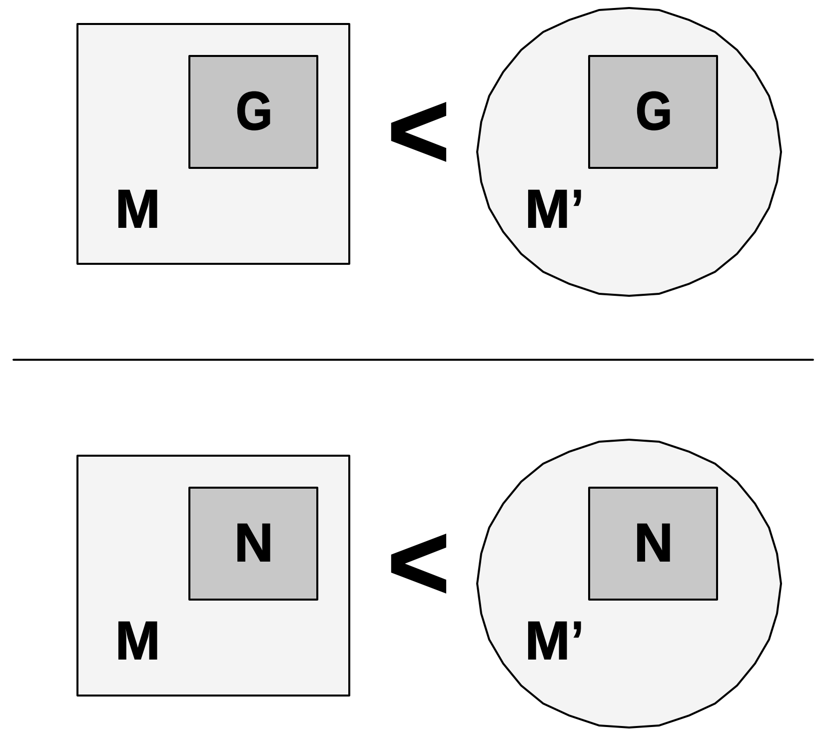
- Constructed solely from the traces of the sub-modes and the mode's transitions.

Refinement

- Defined as usual by inclusion of trace sets.
- Is compositional w.r.t. mode encapsulation.
- Main results: compositional and assume-guarantee rules

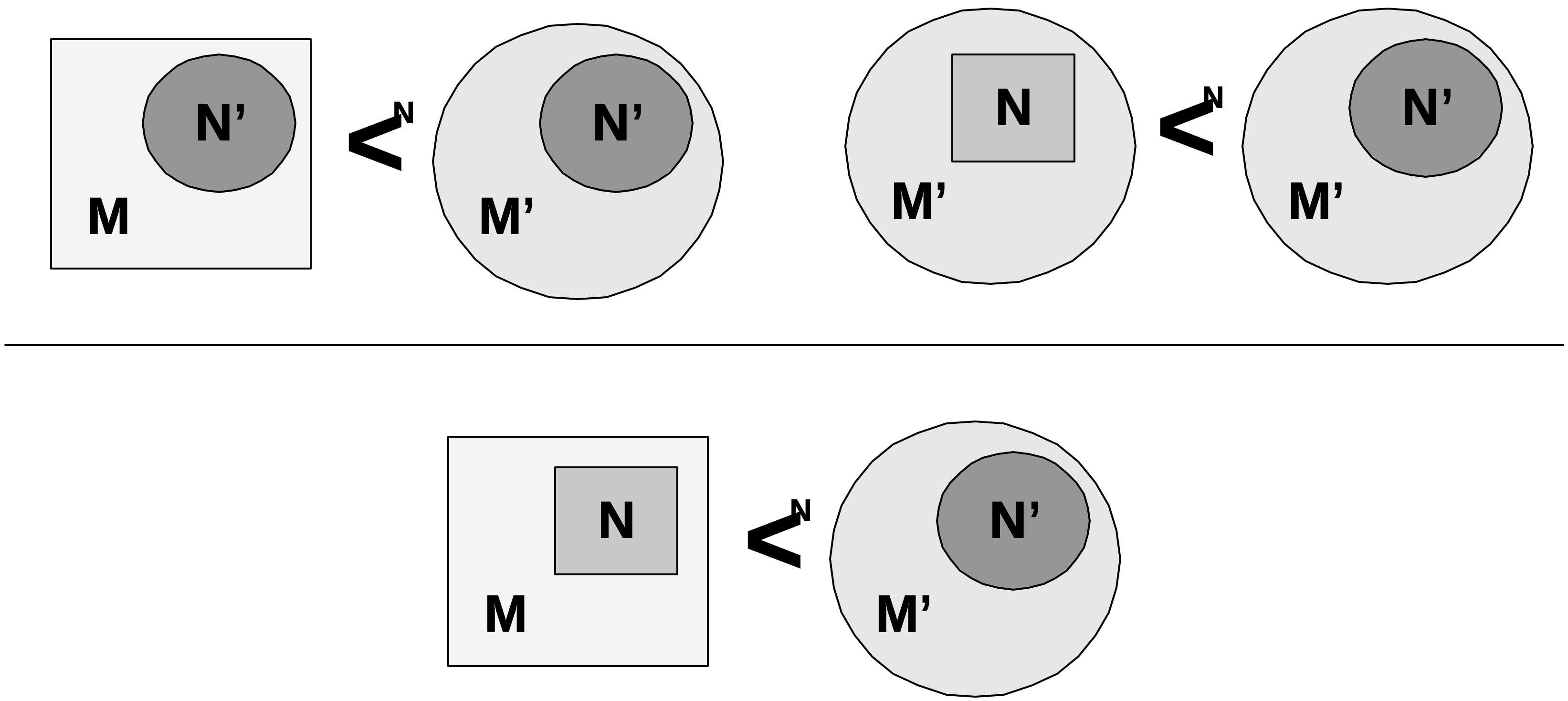
Compositional Reasoning





Sub-mode refinement **Super-mode refinement**

Assume/Guarantee Reasoning



Exploiting Hierarchy in Enumerative Search

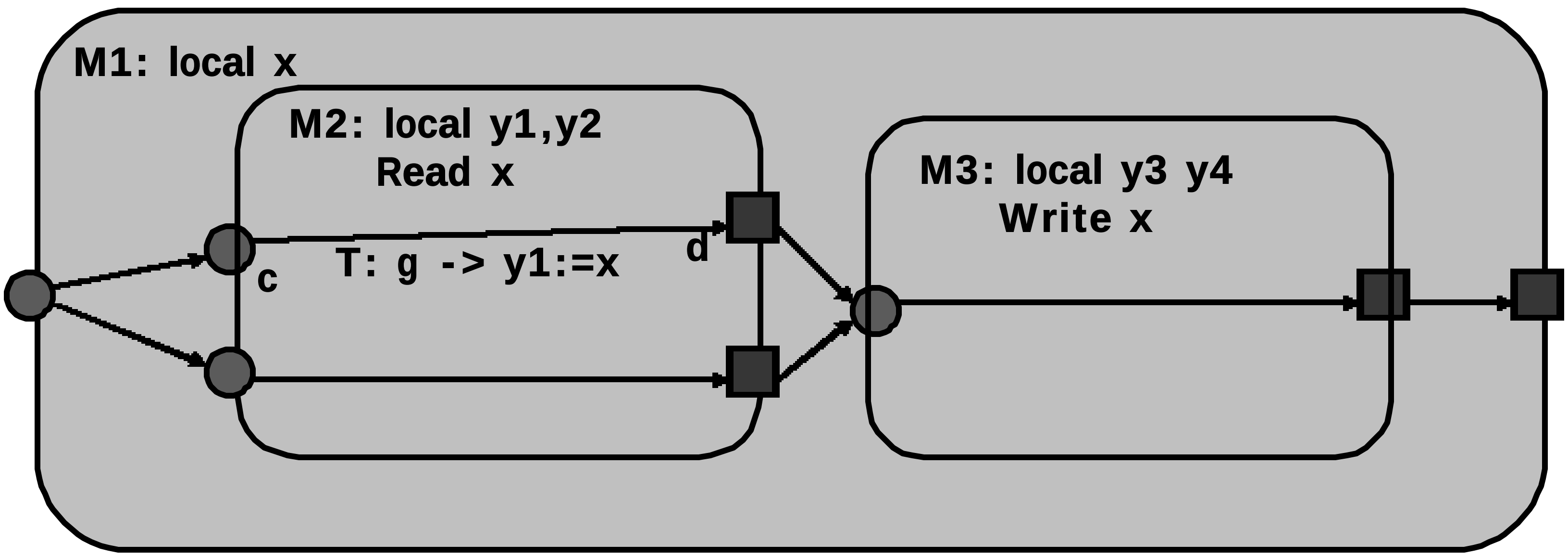
- Local variables do not need to be stored when out of scope
- Hierarchy gives efficient ways of storing state information
- If a mode is used in two places it only needs to be searched once
- Mode's behavior only depends on readable variables - can ignore irrelevant variables

Exploiting Hierarchy in Symbolic Search

- ❑ **Transition relation is indexed by control points**
  - generalization of conjunctively partitioned bdds,
- ❑ **Transition type exploited**
  - for early quantification in the symbolic search,
- ❑ **Reached state space indexed by control points**
  - pool of variables is not global,
- ❑ **Mode definitions are shared among instances.**

Symbolic Search

- Goal: Exploit hierarchical structure in representation and search (avoid flattening)



Transition Relation

- Stored indexed by control points
- Aware of variable scopes

Standard scheme:
T will contribute a conjunct:

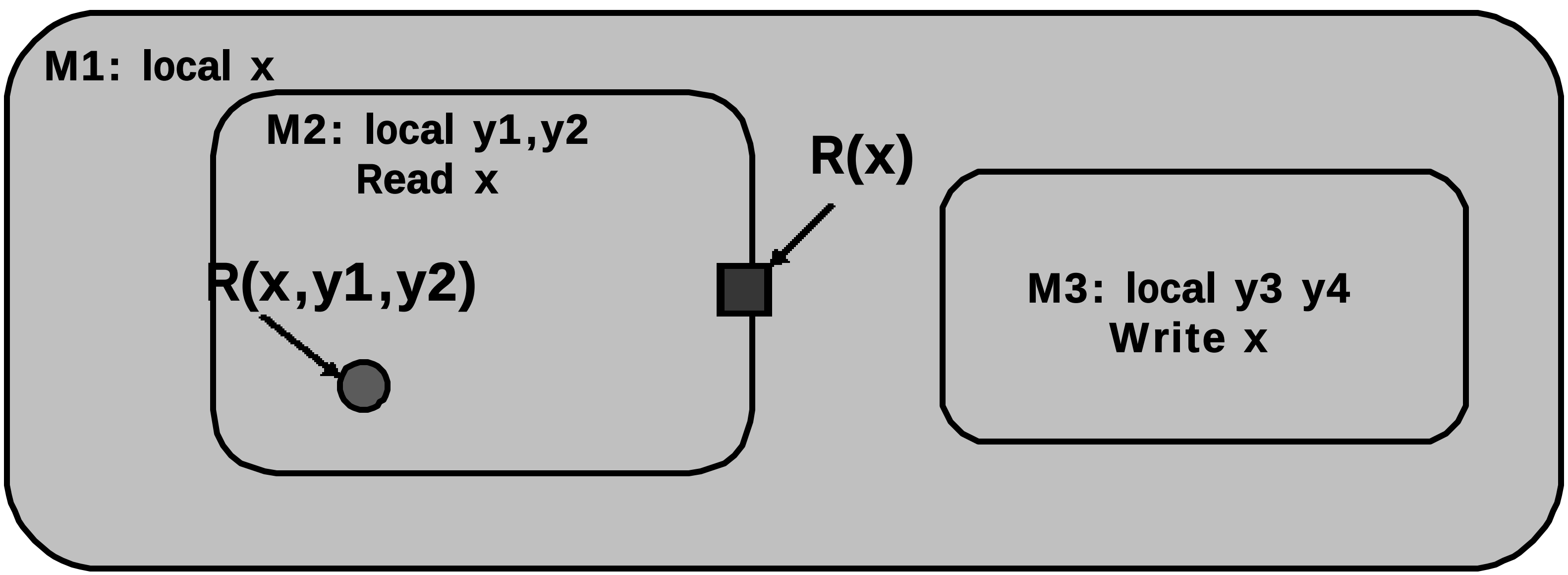
MDD (h=c & g & h'=d & y1'=x & x'=x & y2'=y2 & y3'=y3 & y4'=y4)

Hierarchical scheme:
Transition list indexed by control point c contains:

Target d, MDD (g & y1'=x & y2'=y2)

Reachable Set

- Instead of a global MDD, reachable set is partitioned by control points
- Support set at each point is bounded statically by scoping rules: exploited for quantification



Conclusions

Theoretical study of hierarchy and exploiting hierarchy in verification tools

Acknowledgements

- **Model checking: Yannakakis (FSE 98, TOPLAS 01)**
- **Automata and succinctness: Kannan, Yannakakis (ICALP 99)**
- **Modeling language and semantics: Grosu (POPL 00)**
- **Hermes tool: Grosu, McDougall, Yang (CAV 00,02)**

Current Themes

- **Recursive state machines**
- **Games on hierarchical/recursive structures**