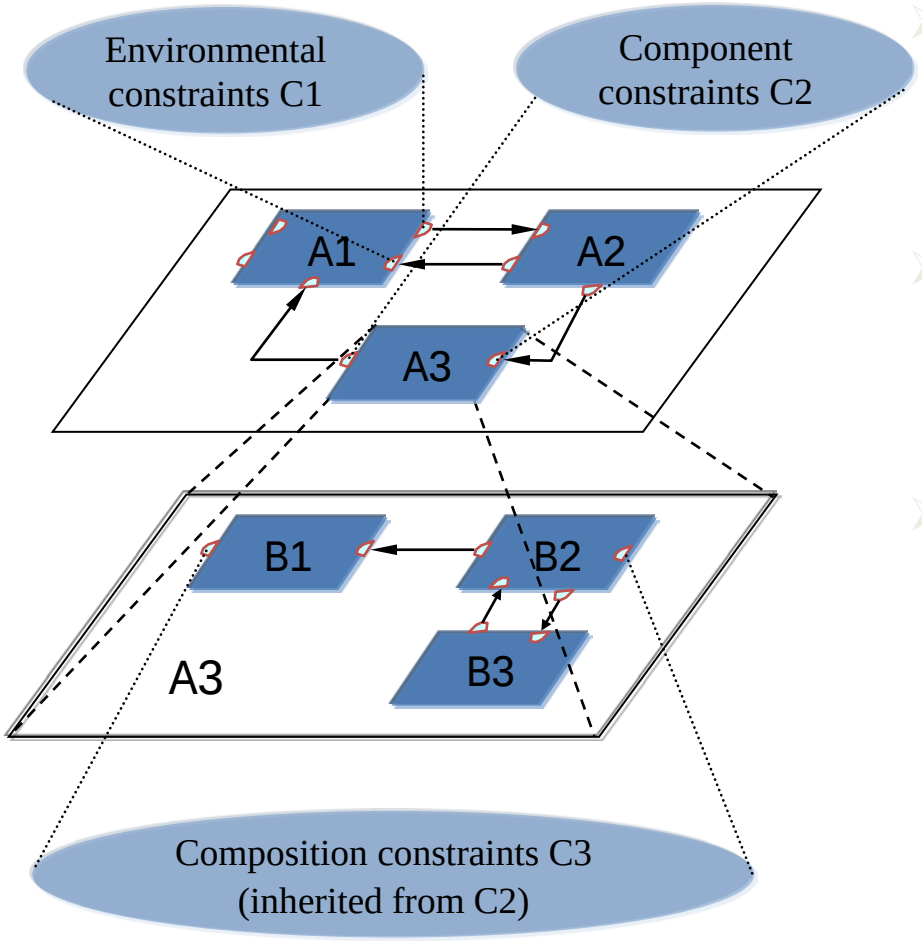
Bounded Model Checking High Level Petri Nets

Xudong He
Florida International University
Miami, FL

Acknowledgements
Funding: *partially supported by NSF grants HRD-0317692 and HRD-0833093*

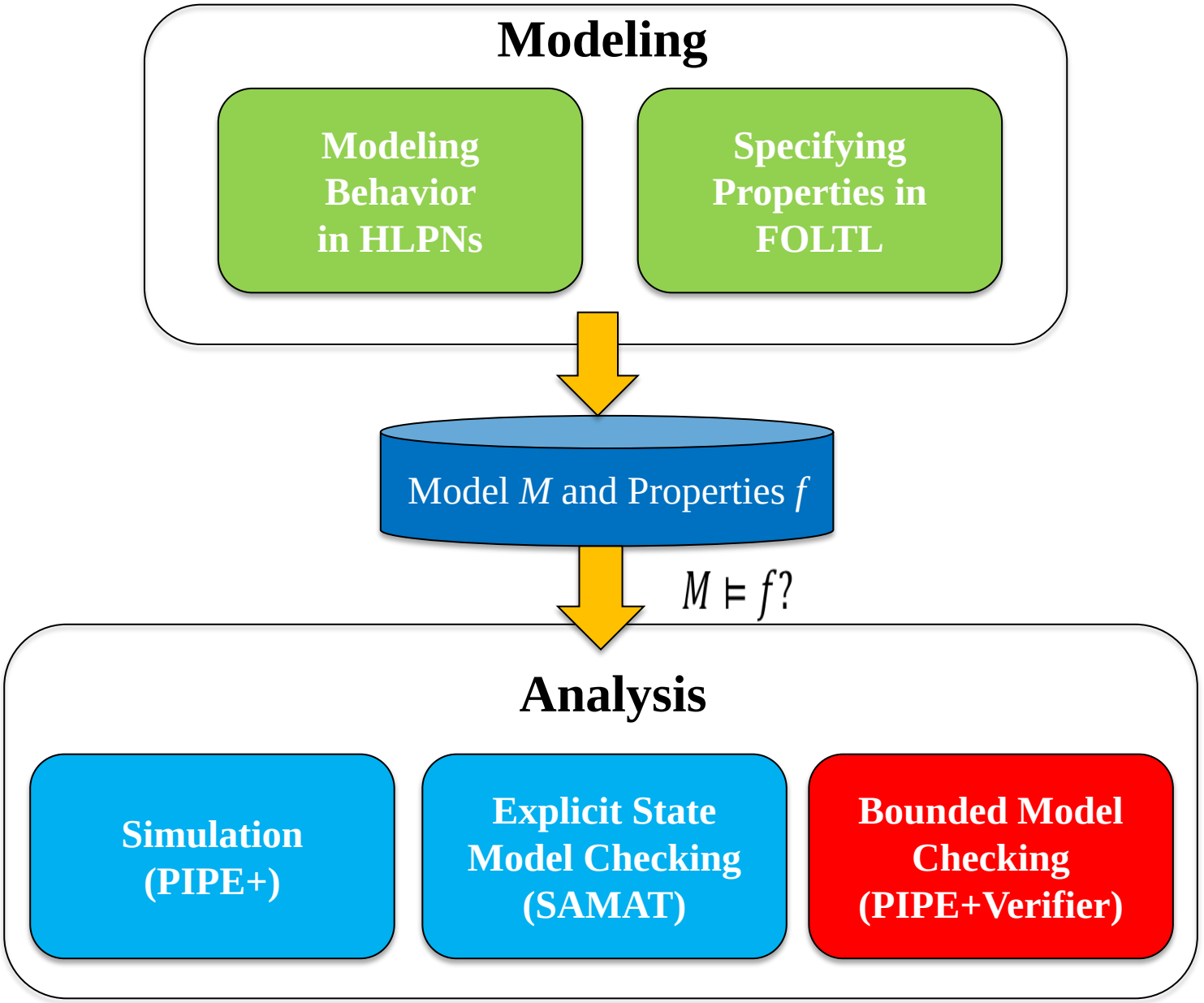
Software Architecture Modeling Methodology (SAM)



- A SAM model $\{C, h\}$
  - A set of compositions C
  - A hierarchical mapping h
- Dual formalisms
  - Petri nets (behavior B)
  - Temporal logic (property S)
- Correctness

  $B \models S$

SAM Modeling & Analysis Tools



High Level Petri Nets (HLPNs)

High level Petri nets (1980s'):

- Syntax (net structure): $N = (P, T, F)$, $P \cap T = \emptyset$, $P \cup T \neq \emptyset$, $F \subseteq P \times T \cup T \times P$
- Static Semantics (net inscription): φ: $P \rightarrow$ *Types*, L: $F \rightarrow$ *Labels*,

  R: $T \rightarrow$ *Logic Formulas* (can be normalized as *pre-cond* $\wedge$ *post-cond*)
- Dynamic Semantics:

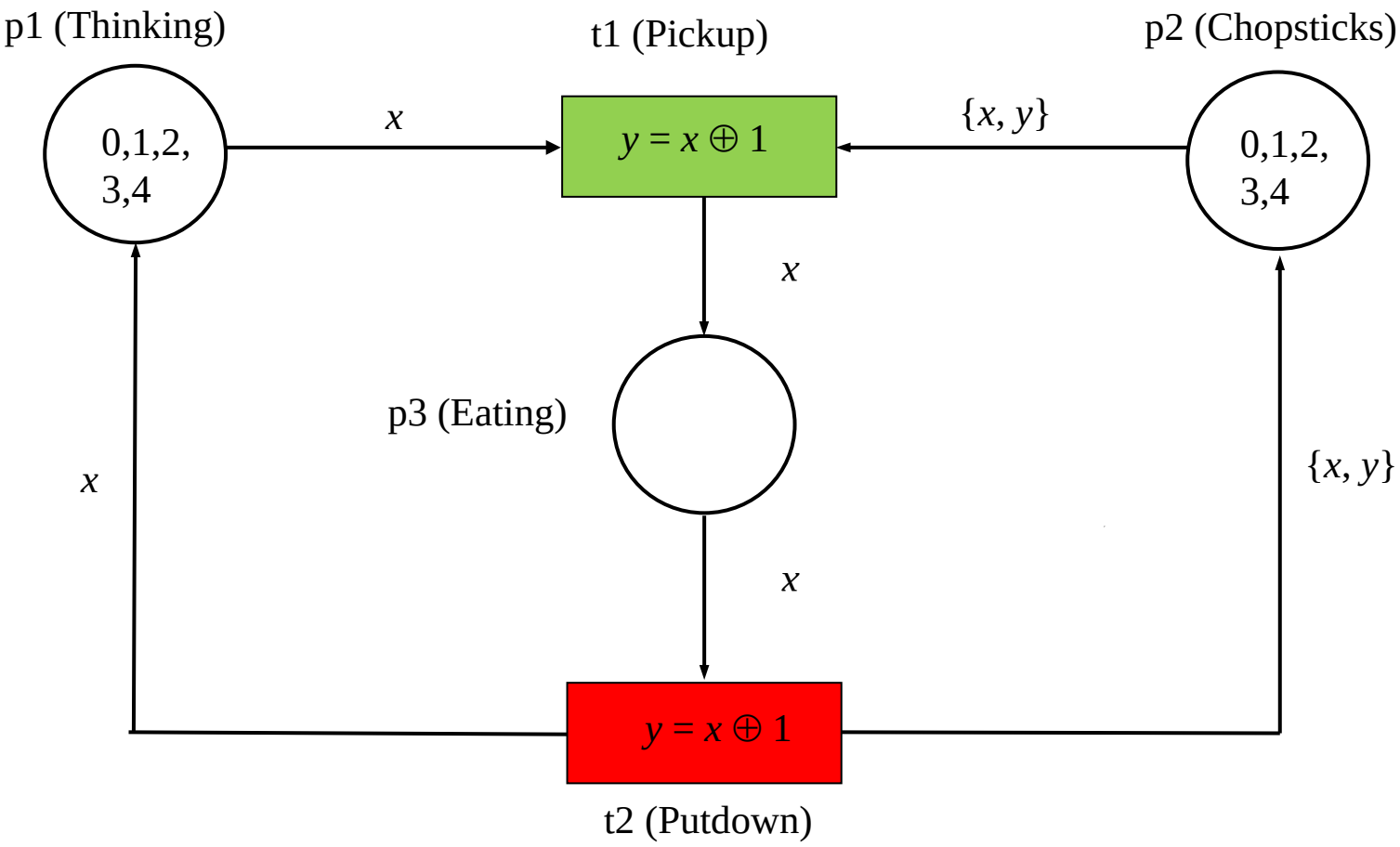
  Initial Marking: M_0: $P \rightarrow$ *Tokens*,

  Transition enabling: $\forall p$: $p \in P.(\bar{L}(p,t):\alpha) \subseteq M(p)) \wedge R(t):\alpha$

  Transition firing: $M'(p) = M(p) - \bar{L}(p,t):\alpha \cup \bar{L}(t,p):\alpha$

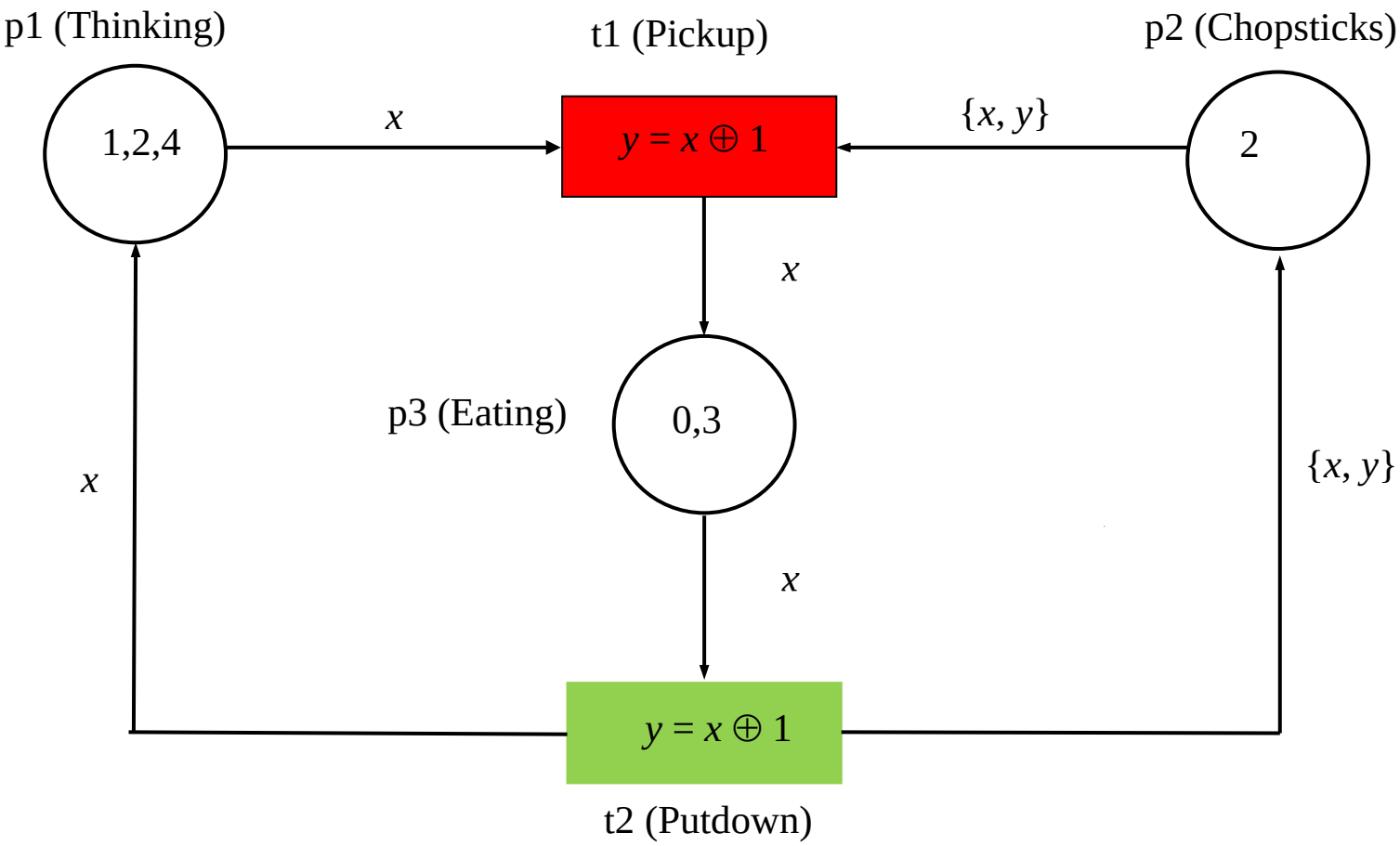
  Execution sequence: $M_0[(T_1, \alpha_1) > M_1[(T_2, \alpha_2) > \cdots M_n[(T_{n+1}, \alpha_{n+1}) > \cdots$
- Expressive power: control structure and flow, data structure and flow, functional processing

High Level Petri Nets – An Example



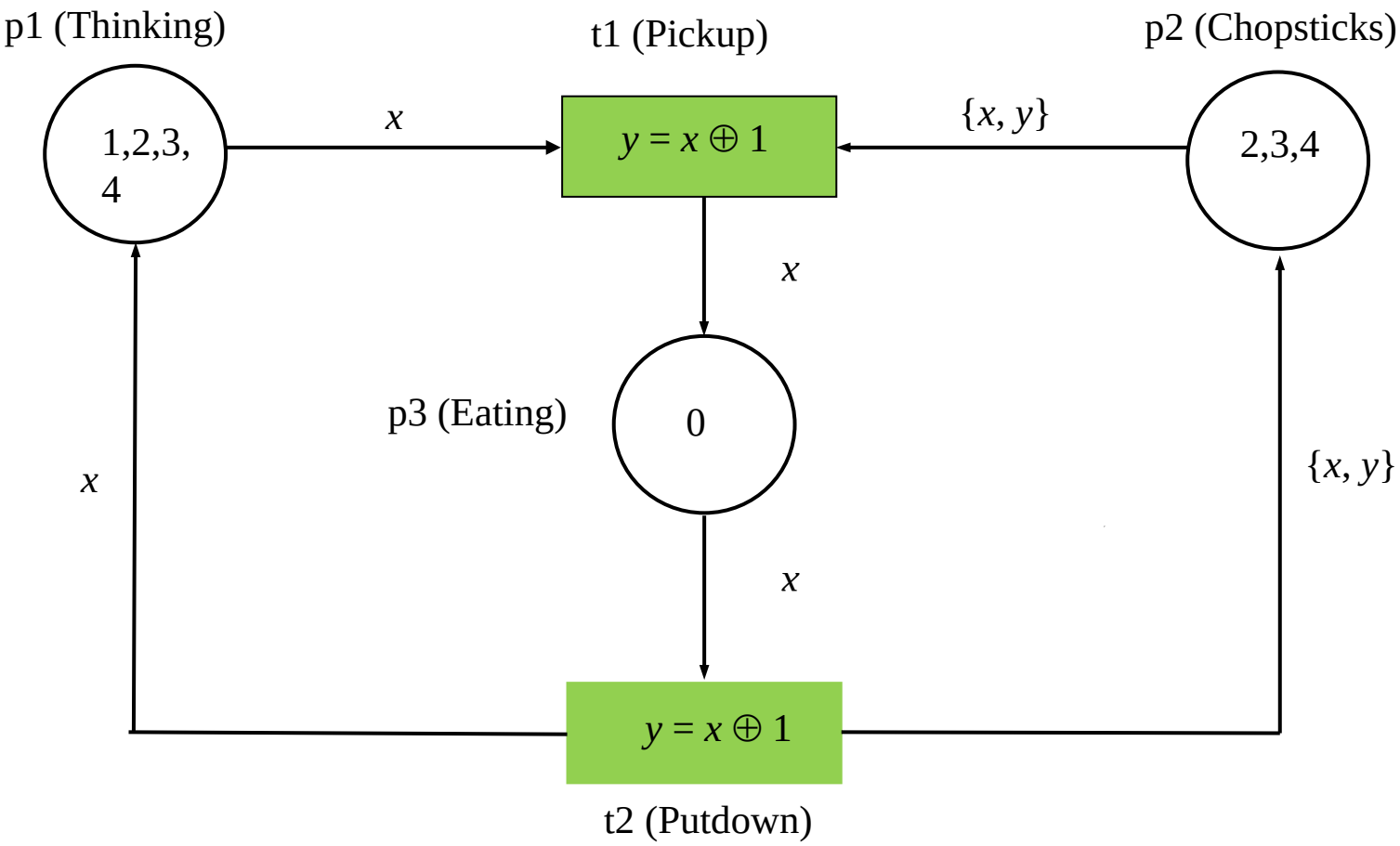
A *HLPN* model of the five dining philosophers' problem

High Level Petri Nets – An Example



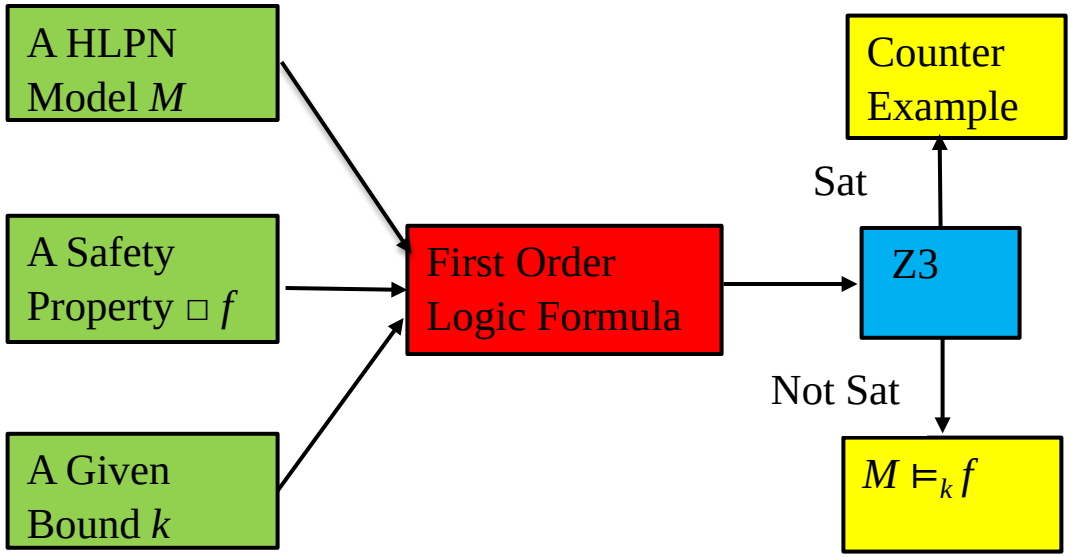
The new marking after firing t1 twice with substitutions
$\alpha 1 = \{ x \leftarrow 0, y \leftarrow 1\}$ and $\alpha 2 = \{ x \leftarrow 3, y \leftarrow 4\}$ concurrently

High Level Petri Nets – An Example



The new marking after firing t2 with substitution $\alpha 3 = \{ x \leftarrow 3, y \leftarrow 4\}$

Bounded Model Checking Process of High Level Petri Nets



Encoding HLPNs for SMT Solver Z3

DEF

$$s : \text{STATETUPLE}$$

ASSERT

$$Initial\_marking(s_0)$$

$$\wedge \bigwedge_{i=0}^{k-1} Transition(s_i, s_{i+1})$$

$$\wedge \bigvee_{i=0}^{k} Negated\_property(s_i)$$

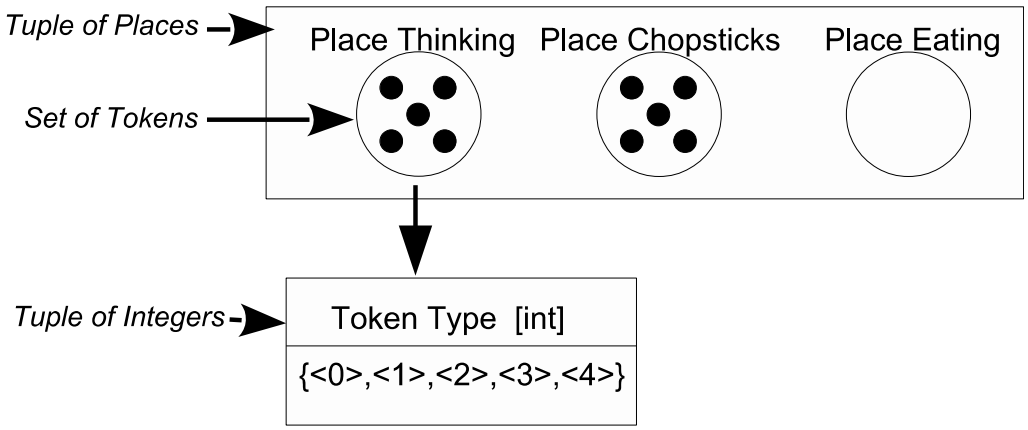
CHECK

- **DEF** defines the global state *s* of a HLPN model;
- **ASSERT** is a first order logic formula encoding an execution sequence of a HLPN up to k+1 states, and the negation of a safety property

Defining STATETUPLE

HLPN Elements	SMT Theory
HLPN Model	Tuple (Places)
Place Type	Set Type (Tokens) **unbounded**
Token Type	Tuple (Integer or String Values)
Primitive Data	Integer or String (Mapping to Integer)

Dining Philosopher Problem in HLPN model



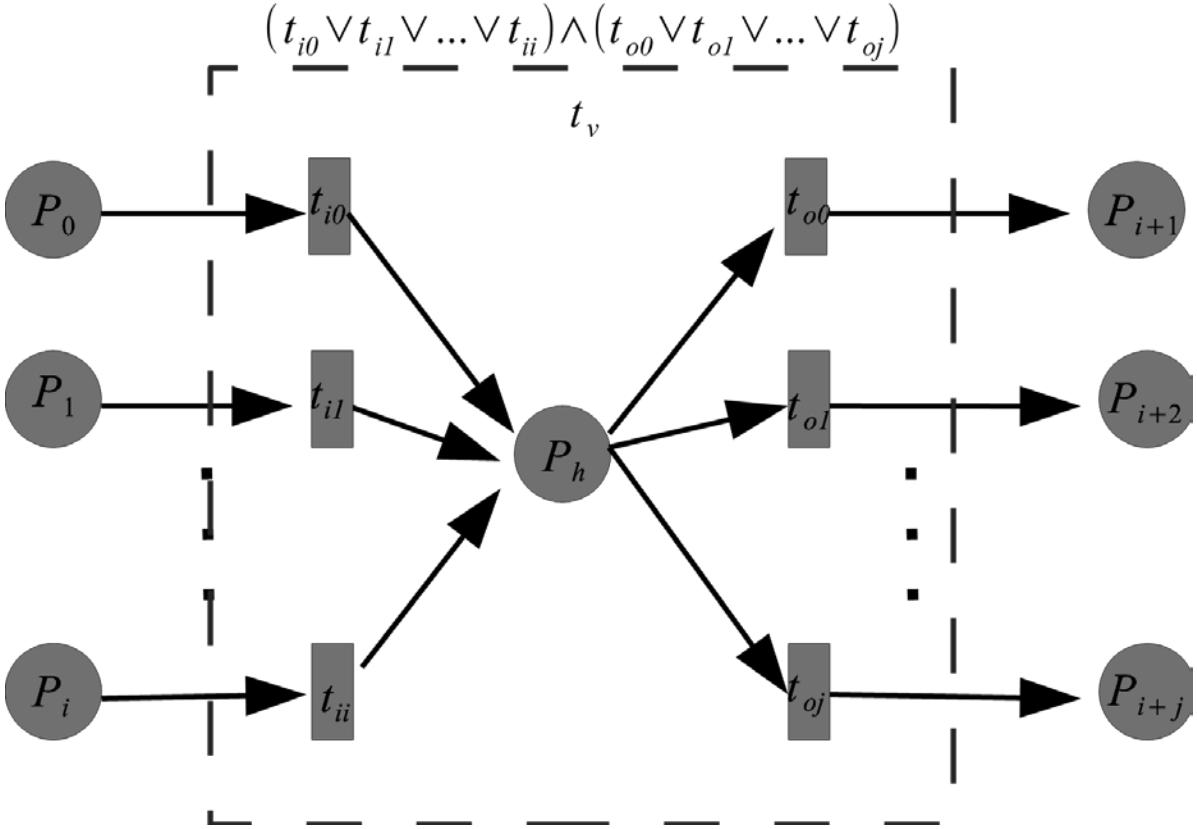
Defining Transition Formula

- Transition(s_i, s_{i+1}) – a disjunction of the transitions *t* in the HLPN model (assume the HLPN model has *n* transitions):

$$Transition(s_i, s_{i+1}) = \bigvee_{j=1}^{n} t_j(s_i, s_{i+1})$$

  - $t_j(s_i, s_{i+1})$ is defined using an if c_0 then c_1 else c_2 structure, where c_0 is the precondition, c_1 is the post-condition and c_2 updates nothing $s_{i+1} = s_i$;
- The above naïve translation captures all possible interleaving, and results in exponential formula size growth;
- By exploring net structure and transition dependencies, we can reduce the size of resulting formula.

Reducing the Size of Transition Formula



- When P_h is neither an initial marking place nor property identified place:

$$T(s,s') = \left(t_{i0}(s,s') \vee t_{i1}(s,s') \vee \ldots\right) \wedge \left(t_{o0}(s,s') \vee t_{o1}(s,s') \vee \ldots\right)$$

Experiments in PIPE+Verifier

- PIPE+Verifier is to check the first three high level Petri net models from the annual Model Checking Contest @ Petri Nets:
  - Dining Philosophers
  - Shared Memory
  - Token Ring
  - Mondex smart card system (the first pilot project of the International Grand Challenge on Verified Software).
- The detailed experiment results are in the Proc. of ICFEM 2014.

Related Work

Name	Petri Net Type	Analysis Technique
ALPiNA	Algebraic Petri Nets	Decision Diagrams
Cunf	Contextual Nets	Net Unfolding, Satisfiability Solving
GreatSPN	Stochastic Petri Nets	Decision Diagrams
ITS-Tools	Timed Petri Nets, ETF, DVE, GAL	Decision Diagrams, Structural Reduction
LoLA	Place/Transition Nets	Decision Diagrams
Marcie	Stochastic Petri Nets	Decision Diagrams
Neco	High Level Petri Nets	Explicit Model Checking
PNXDD	Place/Transition Nets	Net Unfolding, Decision Diagrams, Topological
Sara	Place/Transition Nets	Satisfiability Solving, Stubborn Sets, Topological
CPN Tools	Colored Petri Nets	Explicit Model Checking

Concluding Remarks

- Preliminary results on bounded model checking of HLPNs;
- More Research Issues:
  - How to use net structural patterns to reduce the size of the encoded formula?
  - How to determine the bound k?
  - How to deal with more complex transition constraints that contain quantifiers?

Thank you!