MODEL CHECKING:
Algorithmic Verification III

E. Allen Emerson

Computer Science Department

University of Texas at Austin
`www.cs.utexas.edu/~ emerson/`

Summer School on Model Checking,
Beijing, October 2010

Model Checking

- automatic method for verifying correctness of (finite state) reactive programs
- global state graph defines a *Kripke Structure M*
- correctness specified using temporal logic formula f
- *check* whether M is a *model* of f: $M \models f$
- (efficient) search algorithm inductively calculates the states of M at which f is true using the *fixed-point characterizations* (recursive definitions) of the basic temporal modalities

Exercises

- Establish fixpoint characterizations are correct for
- $EGFp = \nu Z.\mu Y(EX(Y \wedge p) \vee Z)$
- $AFp = \mu Z.p \vee AXZ$
- $EFp = \mu Z.p \vee EXZ$
- $AGp = \nu Z.p \wedge AXZ$

Automata for Basic LTL Modalities

- For each of Fp, Gp, GFp, FGp
  - there is a Buchi automaton w/ 2 states
  - 1st three: deterministic; last: nondeter.

- Fp:: start state s_0, green state s_1
  -alphabet $\Sigma = \{p, \neg p\}$
  -s_0, p enter s_0, /* spin */
  $-s_0, p$ enter s_1 /* flash green */
  -s_1 on p or $\neg p$ enter s_1 /* trapped flashing green */

- Gp:: state s_0, start, green
  In state s_0 on p re-enter s_0, flash *green*;

In state s_0 on $\neg p$ enter s_1;
In s_1 on any input re-enter s_1 but no flash.

- GFp:: s_0 start state; s_1 green state;
  On input p from s_0, s_1 enter s_1, flash green;
  On input $\neg p$ from s_0, s_1 enter s_0, not flash

- FGp:: s_0 start, s_1 green state;
  In s_0 consume all input until (guessed)
  time when all $\neg p$'s seen; nondeter. choose to enter s_1
  In s_1 on p flash green
  In s_1 on $\neg p$, abort

- **Exercise**: Prove automata correct.

Automata and TL

- Uniform Automata framework (cf. [Ku94])
  – modelling, specification, model checking, synthesis
- LTLs translatable into automata [ES83], [WVS83], [VW94]
- Automaton nonemptiness as LTL model checking: [EL85]
  $\mathcal{A}$ is nonempty iff $\mathcal{A} \models EGF\, green$ (recurrence)
- LTL model checking as **recurrence** [VW86]
  $M \models Eh$ iff $M' \models EGF\, green$,
  where $M' = M \otimes aut(h)$.
- LTL model checking as **reachability [SB04]**
  $M \models Eh$ iff $M'' \models EF\, blue$, where $M'' = M' \otimes \mathcal{B}$ where
  $\mathcal{B}$ guesses an accepting cycle in M'.

Reduction of complex to simple

- recurrence $EGFp$ to reachability EFp

- makes it easier to think about

- Question: analogous results for BT, mu-calculus?

Irony

- Model checking reduces complex correctness to simple search. possibly expensive

- Model checking *is* exhaustive testing; it shows both the presence of bugs and their absence, Edsger.

State Explosion

Intractably large state spaces are key obstacle to more widespread application of model checking

| state space | can be exp(lpgm txt) or even infinite

Example: Bank automatic teller network
each teller: 10 local states
100 tellers in network
10^{100} global states

Basic Techniques to Limit State Explosion

Symbolic representation: BDDs, polyhedra

Abstraction: suppress irrelevant detail

- homomorphisms
- simulations
- bisimulations
- symmetry reduction

Other Techniques to Limit State Explosion

- partial order reduction: independent operations allow pruning of spurious interleavings
- on-the-fly: memory efficient; incremental model processing
- compositionality: divide and conquer
- parameterized reasoning: correctness for all sizes n

Abstraction

Establish a correctness preserving correspondence between the original large system and a derived, small and less detailed system

Symmetry Reduction

Symmetry Reduction

- Provides a means of limiting state explosion for systems composed of many interchangeable processes/subcomponents
- Potentially of broad applicability as most systems consist of many copies of a few types of subcomponents
- Often yields exponential savings
- Can be applied automatically

Model of Computation

Structure $M = (S, R)$, where

$S = L^i$: finite set of states

I: finite set of process indices

L: finite set of local process sates

Global state $s = (l_1, ..., l_n)$

$R \subseteq S \times S$: total

M: global state graph of program $P = ||_i K_i$ of many (homogeneous) processes K_i running in parallel

Symmetry Reductions for State Explosion

- Reduce model checking $M = (U)^{(k)}$, large constant k, to model checking quotient structure:
  $\bar{M} = M/\Gamma, \Gamma \leq Aut\, M \leq Sym\, k$

- "State" symmetry(Naive symmetry)
  $M, s_0 \models \bigwedge_i f_i$ iff $M, s_0 \models f_1$

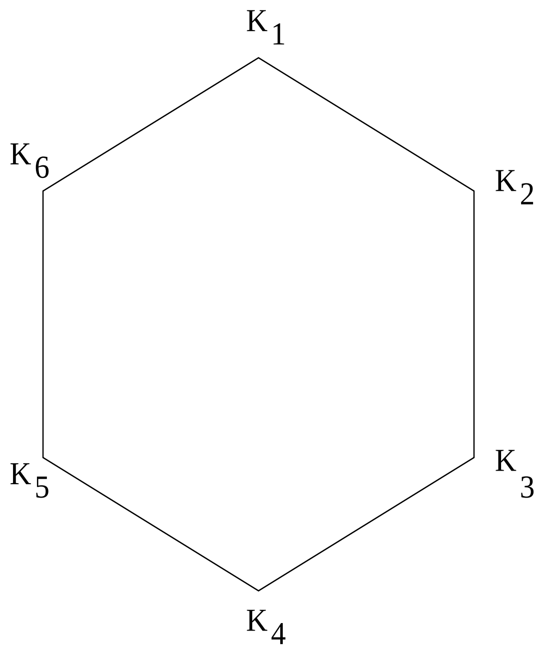
Syntactic and Semantic Symmetry

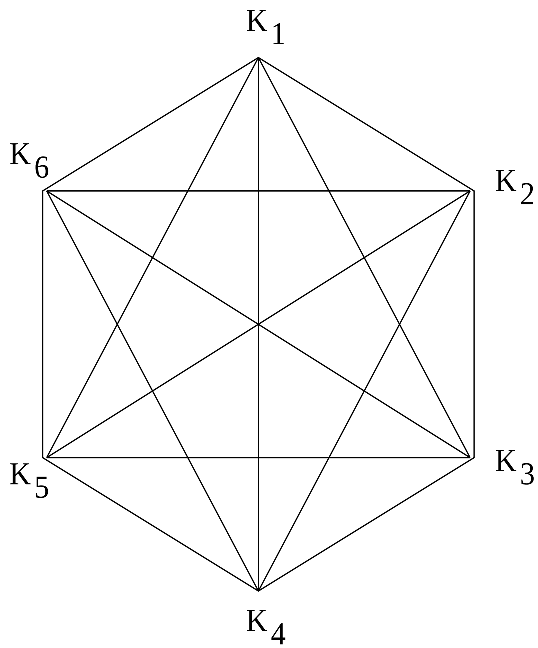
Theorem: For a system with network topology CR, all of whose processes(or subcomponents) are isomorphic and normal,

$$Aut\, CR \leq Aut\, M$$

isomorphic: identical up to reindexing
normal: treat neighbours the same

Intuition: Syntactic symmetry is system description induces semantic symmetry in state graph.





CR: ring of size *n*

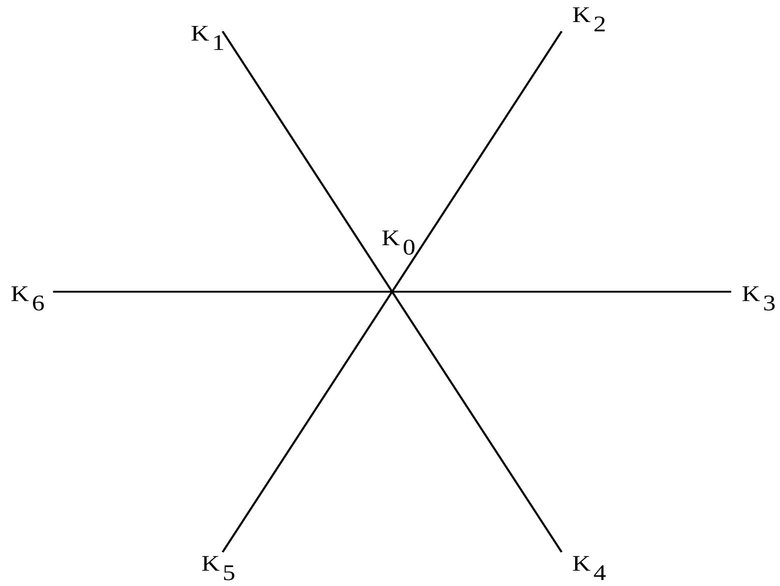
Aut CR = { *n* rotations,
n reflections}

| *Aut* CR| = 2 *n*

CR: complete graph on *n* nodes

Aut CR = *Sym* [1..n]

| *Aut* CR| = *n!*

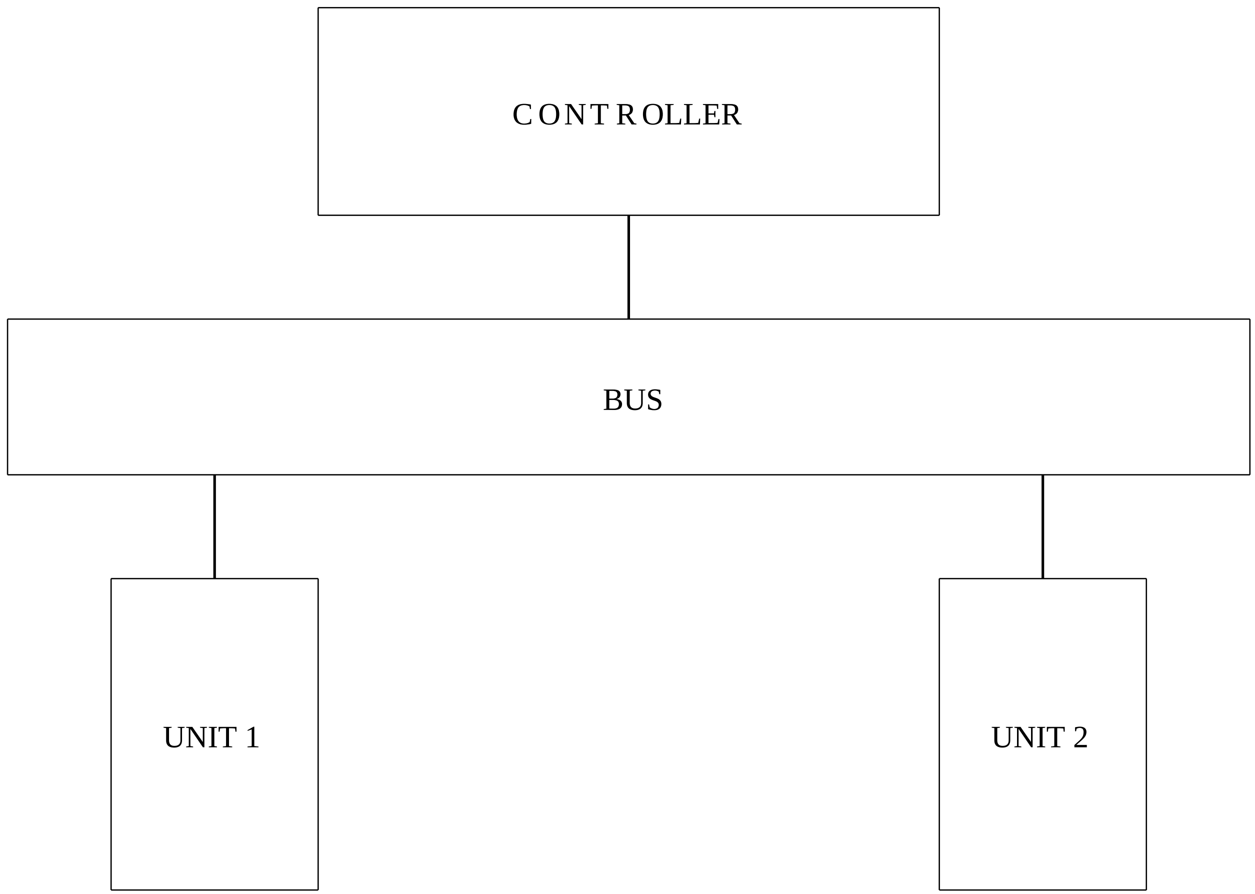


CR: star graph with n arms

$$AutCR = Sym[1..n]$$

$$|AutCR| = n![vs.(n+1)!]$$

Asterisk Graph Model



for homogeneous units

Preliminaries:
Groups & Models

Definition of a Group

Group $\mathbf{G} = (G, \circ)$, $\circ : G \times G \to G$

- associativity: $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$
- identity: $\exists e \in G : (\forall g \in G : (e \circ g = g \circ e = g))$
- inverse: $\forall g \in G : (\exists g^{-1} \in G : (g \circ g^{-1} = g^{-1} \circ g = e))$

Applicable Group Theory: I

$I = [1..n]$: index set

1-1, onto total function $\pi : I \to I$: permutation

$Sym\, I =$ the group of all permutations on I

$G \leq Sym\, I$: subgroup

G acts on S as π applied to s: e.g.,

$S = (N_1\, T_2\, C_3)$, $\pi = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$

$\pi(s) = (N_{\pi(1)}\, T_{\pi(2)}\, C_{\pi(3)}) = (N_3\, T_1\, C_2) = (T_1\, C_2\, N_3)$

Applicable Group Theory: II

An *automorphism* h of $\mathcal{M} = (S, R)$ is a total function $h : S \rightarrow S$ such that

- h is 1-1, onto
- $s \rightarrow t \in R \Rightarrow h(s) \rightarrow h(t) \in R$

$Aut\,\mathcal{M} = \{\pi : \pi \text{ defines an automorphism of } \mathcal{M}\}$
$Aut\,s = \{\pi : \pi(s) = s\}$
$Aut\,f = \{\pi : \pi(f) \equiv f\}$
$Auto\,f = \bigcap\{Aut\,p : p \text{ is a maximal, propositional subformula of } f\}$.

Applicable Group Theory: III

Example: $Aut\, p_1 = \{Id\}$

$$Aut\, p_1 \wedge p_2 = \left\{ \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}, Id \right\}$$

$$Aut\, p_1 \Rightarrow p_2 = \{Id\}$$

Example: $f = \mathsf{E}(\mathsf{F}p_1 \wedge \mathsf{G}p_2))$

$$Auto\, f = Autp_1 \bigcap Autp_1 \wedge p_2 = \{Id\}$$

$$g = \mathsf{E}(\overset{\infty}{\mathsf{F}}\; p_1 \wedge \overset{\infty}{\mathsf{F}}\; p_2)$$

$$Auto\, f = Aut\, p_1 \bigcap Aut\, p_2 = \{Id\}$$

Simple Symmetry

Naive, Naturalistic, or State Symmetry

Simple Symmetry: Basic Idea

Suppose $Aut\, M = Aut\, s_0 = Sym\, I$
ex: $S_0 = (N_1, ..., N_n)$ in mutex solution

Observation: $M, s_0 \models \bigwedge_i g_i$ iff $M, s_0 \models g_1$
($\Rightarrow$): Obvious
($\Leftarrow$): Pick $i \in I$ and $\pi \in Auts_0 \bigcap AutM = SymI s.t. \pi(1) = i$.
$M, \pi'(s_0) \models g_{\pi'(1)}$ for any $\pi' \in AutM$
For $\pi' = \pi$, simplifies to
$M, s_0 \models g_i$.

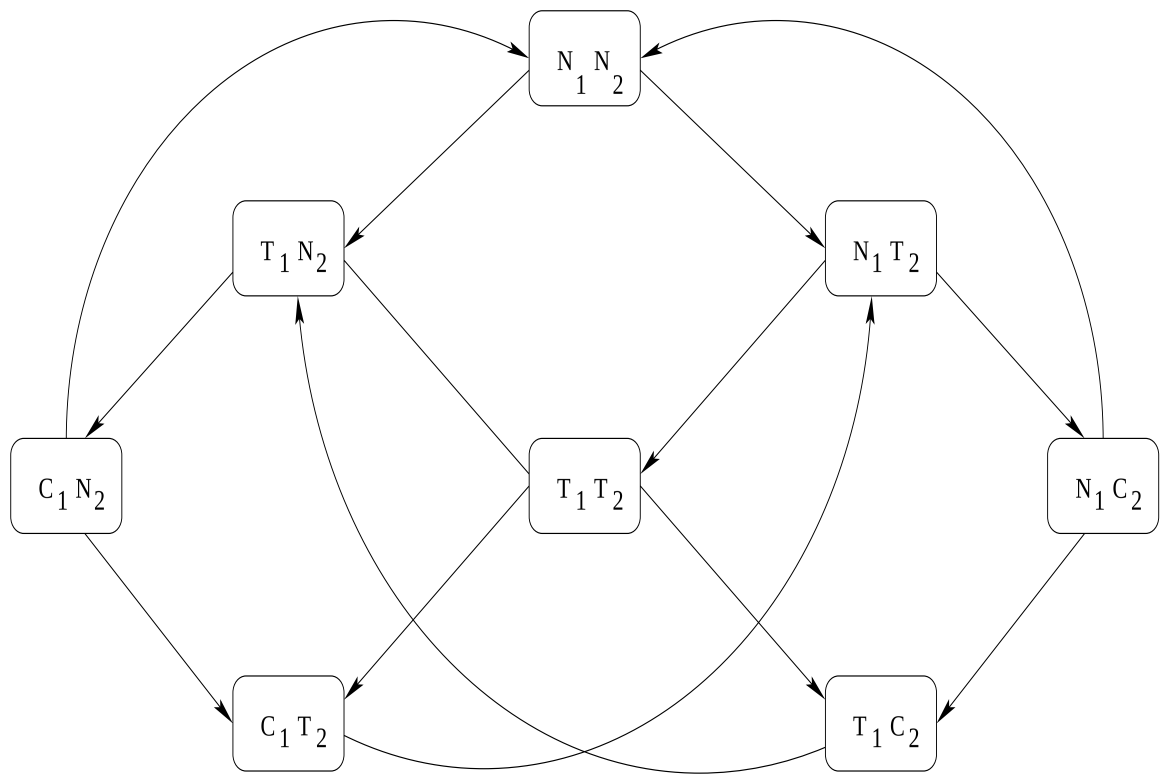
Applications:

- $M, s_0 \models \wedge_{i \in I} \mathsf{AG}(N_i \Rightarrow \mathsf{AF} C_i)$ iff $M, s_0 \models \mathsf{AG}(N_1 \Rightarrow \mathsf{AF} C_1)$
- Used by [Pandey-Bryant] to verify memory arrays
- Used by [McMillan] in Cadence-SMV
- Used throughout distributed systems community

Group-Theoretic Approach

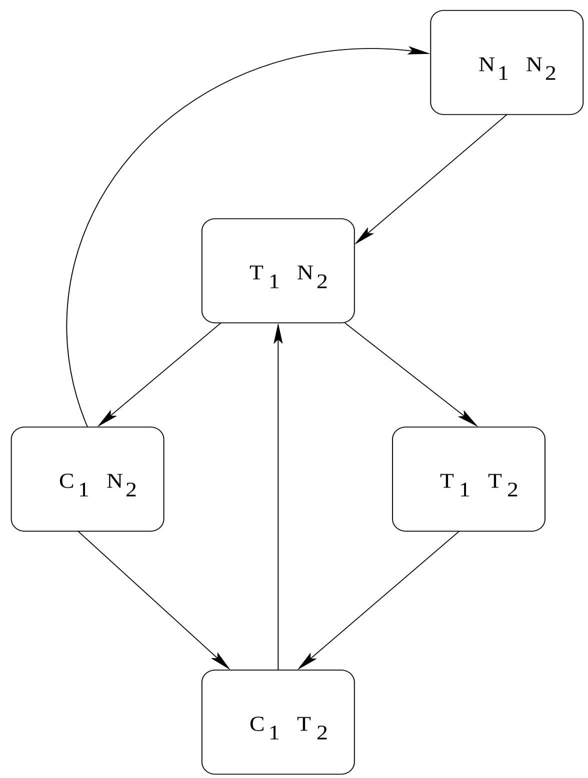
Collapse big graph M by identifying G-symmetric states.

Model check over quotient $\bar{M}$.



$N_1 N_2$
$T_1 N_2$
$N_1 T_2$
$C_1 N_2$
$T_1 T_2$
$N_1 C_2$
$C_1 T_2$
$T_1 C_2$

AG$(\neg(C_1 \wedge C_2))$ preserved



Basic (Rough) Idea

M: global state graph of $||_i K_i$ (typically the K_i are isomorphic

$AutM$: group of automorphisms of M (characterizes symmetry of M

quotient structure $\overline{M} = M/AutM$ (collapsed structure with clusters of symmetric states)

Any path in $\overline{M}$ corresponds roughly to one in M and conversely

Conclusion: $M, s \models f$ iff $\overline{M}, \overline{s} \models f$, where f is any CTL^* formula and $\overline{s}$ is the cluster of s

Quotient Structure

Equivalence relation:: $s \equiv_G t$ iff $\exists \pi \in G : t = \pi(s)$

$[s] : G - orbit$: equivalence class $\bar{s}$: representative of $[S]$

$\bar{\mathcal{M}} \triangleq \mathcal{M}/G \triangleq \mathcal{M}/\equiv_G = (\bar{S}, \bar{R})$, where

$\bar{S}$ is a set of representatives, exactly one for each $[S]$

$\bar{R}$: $\bar{s} \rightarrow \bar{t} \in \bar{R}$ iff

$\exists s' \equiv_G \bar{s}$: $\exists t \equiv_G \bar{t}$: $s' \rightarrow t' \in R$

Advantage: Compression

$\overline{M}$ typically *smaller* than M because representative $\overline{s}$ for equivalence class $[\overline{s}]$ for *many* $s' \in [\overline{s}]$.

Complication: Blurring

The identifying index information of each s' is *muddled*

Compression Theorem

$\mathcal{M}, s \models f$ iff $\mathcal{M}/G, \overline{s} \models f$,
for any $G \leq Aut\,\mathcal{M} \bigcap Auto\,f$ and any f, a formula of

- CTL
- CTL^*
- μ-calculus

Crucial Technical Proviso

- must respect "internal" symmetry of f by <u>not</u> permuting meaning of any maximal propositional subformula

- $Auto\, f =$ group of allowable permutations

- use any Γ subgroup of $Auto\, f \bigcap Aut\, \mathcal{M}$

Examples:
- $Auto\, \mathsf{AG}(\neg(C_1 \wedge C_2)) = \{Flip, Id\}$
  because $Flip \neg(C_1 \wedge C_2) = \neg(C_1 \wedge C_2) \equiv \neg(C_1 \wedge C_2)$
– $Auto\, \mathsf{EF} C_2 = fix\, 2 = \{Id\}$
  because $Flip$ would change 2 to 1, so 2 must be <u>fixed</u>

Why $Auto\,f$?

Consider $\overline{s}$: (C_1, N_2), s : (N_1, C_2)

- $\overline{s}$ can represent s for checking $\mathsf{AG}(\neg(C_1 \wedge C_2))$:

  $$\overline{s} \models \neg(C_1 \wedge C_2) \text{ iff } s \models \neg(C_1 \wedge C_2)$$

  because $\overline{s}, s$ only differ by permutation $Flip$ which leaves the meaning of $\neg(C_1 \wedge C_2)$ unchanged

- $\overline{s}$ <u>cannot</u> represent s for checking $\mathsf{EF}C_2$ because $Flip$ changes the meaning of C_2 to C_1

Automata-Theoretic Approach

An Alternative Automata-Theoretic Approach

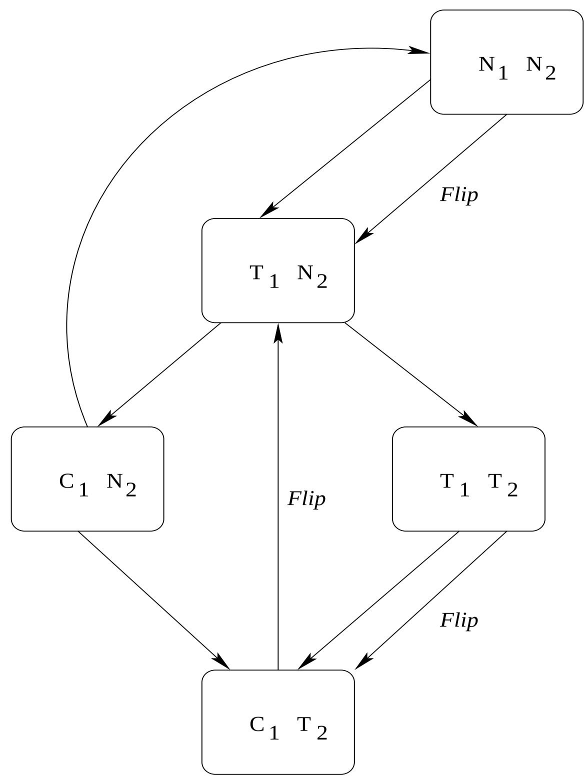
Idea: Work with annotated quotient structure $\bar{M} = M/Aut\,M$ where edges are labelled with permutations indicating how coordinates shift form the representative state to representative state

To check, $M, s \models \mathsf{E} f_i$, where f_i is LTL formula with propositions involving solely index i

Let A be Buchi automaton for f_i

Let $B = A \times I$ be automaton which mimics A but also reads edge permutations to keep track of shifting locations of index i

Check whether $\bar{M} \times B$ is nonempty



N1 N2
Flip
T1 N2
C1 N2
Flip
T1 T2
Flip
C1 T2

Advantages of Automata-Theoretic Approach

- $Auto\, f$ eliminated

- Use one $\bar{M}$ for many f

- Fixed $G \leq Aut\, M$ implies greater compression likely

- handle $\bigwedge_i \mathsf{E} f_i, \bigvee_i \mathsf{E} f_i$ fast