Symmetry Reductions.

A. Prasad Sistla
University Of Illinois at Chicago
Model-Checking

Approach
- Build the global state graph
- Algorithm to check correctness

Applications
- Concurrent Programs.
- Protocols.
- Circuits.
Bottleneck: State Explosion

Has only been used for small size problems.

- number of states grows exponentially.

Techniques to contain state explosion

- Symbolic Model Checking (BDDs)
- Stubborn Sets/Sleep Sets
- Symmetry
  (Due to identical/similar processes)
Outline

- **Symmetry & Quotient Structure**
  - Program Symmetry
  - Formula Symmetry
  - Quotient Structure
  - State Symmetry
- **Annotated Quotient Structure.**
  - Fairness

- **SMC – An implemented system**

- **Reduced Symmetry & Assymetry.**
  - Guarded quotient Structure.
  - Formula Decomposition
  - Subformula tracking.
Model : Shared Variable

**Notation**
- Variable name- $X_{i,j}$
  (denotes a variable shared by processes $i, j$)

**Program** : A set of processes.
**Process** : A set of guarded commands.

**Process $K_i ::**
- $[ \text{LC}_i = 0 \land F_{i,i+1} = 1 \rightarrow F_{i,i+1} := 0, \text{LC}_i := 1; \]
- $\text{LC}_i = 1 \land F_{i,i-1} = 1 \rightarrow F_{i,i+1} := 0, \text{LC}_{i,i+1} := 2; \]
- $\text{LC}_i \rightarrow F_{i,i+1} := 1, F_{i,i+1} := 1, \text{LC}_i := 0; ]$

**Program $K :: K_1 || K_2 || \ldots || K_n$**
Program Symmetry

- $\mathcal{K} :: \mathcal{K}_1 \parallel \mathcal{K}_2 \parallel \ldots \parallel \mathcal{K}_n$
- $\parallel$ is commutative and associative
- $\mathcal{I} = \text{Index set} = \{1, 2, \ldots, n\}$
- For a permutation $\pi$ on $\mathcal{I}$,
- define $\pi(\mathcal{K}_i)$ - Process obtained by changing indices of variables according to $\pi$
  ($X_{i,j}$ changed to $X_{\pi(i), \pi(j)}$)

- $\text{Aut } \mathcal{K} = \{ \pi \mid \pi(\mathcal{K}) = \mathcal{K} \}$
Global State $s$: Assignment of values to variables.

**Global State Graph**: $\mathcal{M} = (S, R)$

- $S$ - Set of global states
- $(s, t) \in R$ iff $t$ is obtained from $s$ by executing a single step of some process.
  (interleaved semantics)

Interested in Symmetry of $\mathcal{M}$:
s: global state.

\( \pi \): permutation on \( \mathcal{I} \),
- \( \mathcal{I} = \{0, 1, \ldots, n-1\} = \) Process Indices.

\( \pi(s) \) is a global state in which
- variable \( X_{\pi(i), \pi(j)} \) gets the value of \( X_{i,j} \) in \( s \)

\( \text{Aut}(\mathcal{M}) = \{\pi | \pi(\mathcal{M}) = \mathcal{M}\} \)
- \( \text{Aut} \mathcal{M} \) is a group

Lemma: \( \text{Aut} \mathcal{K} \subseteq \text{Aut} \mathcal{M} \)
Examples:

- For Symmetric solution of Dining Phil. Problem:
  \[ \text{Aut } \mathcal{M} = \{ \pi : \pi \text{ is a circular perm.} \} \]
  \[ \text{Aut } (\mathcal{K}) = \text{Aut } (\mathcal{M}) \]

- For the Resource controller Prog:
  \[ \text{Aut } \mathcal{M} = \{ \pi : \pi(0) = 0 \} \]
  \[ \pi \text{ can permute the users.} \]
  \[ 0 \text{ is the controller process} \]
Logics of Programs

CTL*:

Temporal operators: $F, G, X, U$

- $F P$ : eventually $P$
- $G P$ : always $P$
- $X P$ : nexttime $P$
- $P U Q$ : $P$ until $Q$

Path Quantifiers:

- $A$ – for all paths
- $E$ – for some path
- $AG(P)$ : Invariance
- $AF(P)$ : Inevitability of $P$

$P$ : Basic assertion, uses indexed variables.

Ex: $LC_i = L$, $X_{ij} > 0$
$f$: a formula in CTL*

$\pi(f)$ obtained by changing indices of variables according to $\pi$.

**Symmetry of formulas:**

$\text{Aut } f = \{\pi : \pi (f) \equiv f\}$

We use a subgroup of $\text{Aut } f$ called $\text{Auto } f$.

$\text{Auto } f = \cap (\text{Aut } p)$

$p$ is a maximal prop. subformula of $f$. 
Examples:

\[ f = AG \left( (T_1 \lor T_2) \rightarrow AF(C_1 \lor C_2) \right) \]

Global liveness for a mutual exclusion problem with two processes.

T_i: Process i is in trying mode.

C_i: Process i is in critical section.

Auto(f) = Sym I = Set of all permutations. \( I = \{1, 2\} \)

\[ g = \bigwedge_{i=1,2} AG \left( T_i \rightarrow AF C_i \right) \]

Aut(g) = Sym I, \hspace{1cm} Auto(g) = \{Id\}
Quotient Structure:

\[ M = (S, \mathcal{R}) - \text{structure} \]

\[ f - \text{CTL}^* \text{ formula} \]

\[ G \subseteq \text{Aut } M \cap \text{Auto } f \]

Equivalence relation \( \equiv_G \) on \( S \)

\[ s \equiv_G t \iff \exists \pi \in G \text{ such that } \pi(s) = t \]

\[ M/G = (S^+, \mathcal{R}^+) : \text{Quotient Structure.} \]

\( S^+ \) has one representative for each equiv. class.

\[ (s^+, t^+) \in \mathcal{R}^+ \iff \text{for some } s^i \equiv_G s^+, \ t^i \equiv_G t^+ \]

\[ (s^i, t^i) \in \mathcal{R} \]
Correspondence Lemma

- There is a bidirectional correspondence between paths of $\mathcal{M}$ and $\mathcal{M}/G$

$$(s_0, s_1, \ldots, s_i \ldots) \in \mathcal{M} \Rightarrow (s_0^+, s_1^+, \ldots, s_i^+ \ldots) \in \mathcal{M}/G$$

$$(s_0^+, s_1^+, \ldots, s_i^+ \ldots) \in \mathcal{M}/G \Rightarrow \forall s_0^\equiv_G s_0^+ \text{, } \exists \text{ path in } \mathcal{M}$$

$$(s_0^\equiv, s_1^\equiv, \ldots, s_i^\equiv \ldots) \text{ such that } s_i^\equiv \equiv_G s_i^+$$
Main Theorem

- For any \( s \in S \),
- \( M, s \models f \iff M/G, s^+ \models f \)
- (Ip & Dill 93, CFG 93, ES 93)

Proof: Uses induction on \( f \) and the corr. Lemma.

Examples

Dining Phil. Problem

- \( f = AG( EX True) \) (absence of deadlock)
- Auto \( f = \text{Sym } \mathcal{I} \)
- Auto \( M \cap \text{Auto } f = \text{all circ. permutations} \)
Example

Two process Mutual Excl. $AG(\neg(C_1 \land C_2))$
Example contnd.

Two process Mutual Excl. $AG(\neg (C_1 \land C_2))$
Quotient Structure.

\[ N_1 N_2 \ t=0 \]

\[ T_1 N_2 \ t=1 \]

\[ C_1 N_2 \ t=0 \]

\[ T_1 T_2 \ t=1 \]

\[ C_1 T_2 \ t=0 \]
Fairness

- Correctness under group fairness is preserved

- $G \subseteq \text{Aut} \mathcal{M} \cap \text{Auto} f$

- Define index $i \equiv j \iff \exists \pi \in G$ such that $j = \pi(i)$

- $C_1, C_2, \ldots, C_k$ are the equivalence classes of indices

- Group fairness: for $\ell = 1, \ldots, k$ some process $\in C_\ell$ executed infinitely often
Fairness Theorem

- \((\mathcal{M}, s)\) satisfies \(f\) under group fairness \iff\ 
  \((\mathcal{M}/G, s^+)\) satisfies \(f\) under group fairness

- Example:

  \[f = AG ((T_1 \lor T_2) \rightarrow AF (C_1 \lor C_2))\]
  
in the mutual exclusion example.
Incremental Computation of $\mathcal{M}/G$: $\mathcal{M} = (\mathcal{S}, \mathcal{R})$

$\mathcal{S}^+ = \{ s_0 \}$, $s_0$ - init. State

$\mathcal{Q} = \{ s_0 \}$

While $\mathcal{Q} \neq$ empty

\begin{align*}
    \text{s:= dequeue ( } \mathcal{Q} \text{ );} \\
    \text{for each successor t of s} \\
    \quad \text{if ( } \exists u \in \mathcal{S}^+ \text{ such that } u \equiv_G t \text{ )} \\
    \quad \quad \text{then add ( s, u) to } \mathcal{R}^+ \\
    \quad \text{else } \mathcal{S}^+ = \mathcal{S}^+ \cup \{ t \}, \\
    \quad \quad \text{add (s, t) to } \mathcal{R}^+
\end{align*}

end for.

end while.

Checking $u \equiv_G t$ is a difficult problem.
Savings in the size of state space.

We can obtain exponential savings in some cases.

Resource Controller problem:

\[ n - \text{# of users} \]

\[ m - \text{# of states of the controller} \]

Assume each user has 3 states

\[ \mathcal{M} \text{ has } \mathcal{O}(m \cdot 3^n) \text{ states.} \]

\[ \mathcal{M}/G \text{ has } \mathcal{O}(m \cdot n^3) \text{ states} \]
Finding suitable G

- Largest possible $G \subseteq \text{Aut } \mathcal{M} \cap \text{Auto } f$
  gives maximum compression.

- **Difficulties:**
  - Computing $\text{Aut } \mathcal{M}$ is difficult, as hard as graph isomorphism
    - use $\text{Aut } \mathcal{K}$ (can be determined from syntax)
  - Computing $\text{Auto } f$ can be hard
  - Many times $\text{Aut } \mathcal{K}, \text{Auto } f$ are known in advance
  - For isomorphic processes, $\text{Aut } \mathcal{K} = \text{Aut } \mathcal{CG}$
Automata Theoretic Approach & AQS

Don’t need to consider formula symmetry.

Use an Annotated Quotient Structure. (AQS)

Take $G \subseteq \text{Aut} \ M$

edges are labeled with permutations
if $(s^+, t) \in \mathcal{R}$ then
there is an edge from $s^+$ to $t^+$
labeled with $\pi$ where $t = \pi(t^+)$
Correspondence between $\mathcal{M}$ & $\mathcal{M}^+$

Converse also holds

$\mathcal{M}^+$ is a succinct encoding of $\mathcal{M}$. 

$\pi_1 \pi_2 \pi_3 (s_3^+) \\
\pi_1 \pi_2 (s_2^+) \\
\pi_1 (s_1^+) \\
s_0^+$

unwind(x)

$\pi_1 \pi_2 \pi_3 (s_3^+) \\
\pi_2 (s_2^+) \\
\pi_1 (s_1^+) \\
s_0^+$

$\pi_3 \\
\pi_2 \\
\pi_1 \\
s_3^+ \\
s_2^+ \\
s_1^+$

$\mathcal{M}$

$\mathcal{M}^+$

$x$
Annotated Quotient:

\[ G = \{ \text{Id,Flip} \} \]
Correctness under fairness.

Want to check if $\mathcal{M}, s_0^+ \models E(\phi \land f_i)$

where $\phi$ expresses weak fairness. $A$ – Automaton for $f$.

Defn: $\mathcal{B}^+ = \mathcal{M}^+ \times A \times I$

$$(s^+, q, i) \xrightarrow{\pi} (t^+, r, j) \iff$$

$s^+ \xrightarrow{\pi} t^+ \in \mathcal{M}^+$,

$q^+ (s^+\downarrow i) \xrightarrow{} r \in A$ and

$\pi^{-1}(i) = j$

Defn: A SCC $C^+$ of $\mathcal{B}^+$ is green

if $\exists (s^+, q, i) \in C^+$ such that $q \in$ GREEN.
Theorem:
\[ M, s_0^+ \models E(\phi \land f_i) \iff B^+ \text{ contains a subtly fair and green SCC } C^+ \text{ that is reachable from } (s_0^+, q_0, i). \]

How to check if \( C^+ \) is subtly fair?
use a threaded graph \( C^* \).
Alg to mark all states in $\mathcal{M}$ that satisfy: $E(\phi \land f_i)$

1. Construct $\mathcal{A}$
2. Construct $\mathcal{B}^+ = \mathcal{M}^+ \times \mathcal{A} \times \mathcal{I}$
3. For each SCC $C^+$ of $\mathcal{B}^+$
   - Check if $C^+$ is subtly fair
   - Construct $C$
   - Check if $C^*$ is plainly fair.
4. For each $s^+ \in \mathcal{M}$
   - mark $s^+$ if a subtly fair and green SCC is reachable from $(s^+, q_0, i)$ in $\mathcal{B}^+$
Complexity: $O(|M^+|.|A|.n^2)$

Above approach can be extended to strong fairness

Complexity: $O(|M^+|.|A|.n^3)$

On-the-Fly Algorithm is more subtle.
Implementation: SMC
(Symmetry based Model – Checker)

- Developed at Univ. of Illinois at Chicago
- Uses AQS based approach
- Employs a variety of symmetries:
  - Program symmetry, State symmetry
- Uses variety of on-the-fly options
  - (AQS and/or product structure constructed on-the-fly)
Implementation : SMC(cont’d)

- Allows different fairness specifications (weak/strong features)

- Used for checking real world examples, found bugs in the *Fire-Wire* protocol.
Reduced Symmetry

- Symmetric system with asymmetric constructs.
  Ex: resource controller with priorities.

- Partially Symmetric system.

Introduce

1) Guarded Quotient Structures (GQS)
   Further extension of AQS

2) Two new techniques
   (a) Formula decomposition
   (b) Sub-formula tracking
Guarded Quotient Structures (GQS)

$G = (S, E)$ - reachability graph.

$\text{Aut}(G)$ – group of symmetries/automorphisms of $G$.

$\text{Aut}(G)$ – may be small. Not much compression

Add edges to $G$ and obtain an expanded graph

$H = (S, F)$ so that

$F \supseteq E$

$\text{Aut}(H) \supseteq \text{Aut}(G)$
GQS – contd.

Construct the AQS of H

Add an edge condition with each edge of H (called guards)

Used during the unwinding process.

The resulting structure is GQS(G)
Example

Mutual Exclusion with priority for Process 1 (The Graph G)
Example

Mutual Exclusion with priority for Process 1
GQS for the system with priority.
Unwinding:

**GQS:**

φ,gi

s

t

**GQS-Struct**

\[(t, \phi^{-1}(P_1), \ldots, \phi^{-1}(P_k), \phi^{-1}(\theta_1), \ldots, \phi^{-1}(\theta_\ell))\]

\[(s, P_1, \ldots, P_k, \theta_1, \ldots, \theta_i, \ldots, \theta_\ell)\]

Correspond to & track atomic predicates in the formula

Track edge conditions

Correspond to & track edge
To Check $f$

- Unwind the GQS to get GQS-struct
  - unwinding done w.r.t. $P_1, P_2 \ldots P_k$ and the edge conditions.
  - use the edge conditions to consider only those edges in $G$.
- Check $f$ in GQS-struct
- When optimized: GQS-struct has
  - no more nodes than as $G/\text{Aut}(G)$
  - fewer nodes in many cases

Important advantage of GQS:
- Can use formula decomposition
- Subformula tracking
Suppose $f = f_1 \land f_2$ \ ($f_1, f_2$ are state formulas)

Only $P_1$ appears in $f_1$

Only $P_2$ appears in $f_2$

Unwind GQS w.r.t. $P_1$ and check $f_1$

Unwind GQS w.r.t. $P_2$ and check $f_2$

Avoids unwinding w.r.t $P_1$ and $P_2$ simultaneously.
Generalization.

- Group Top level subformulas into classes
  - all subformulas in a class contain same atomic propositions
  - Sets of atomic propositions of different classes are disjoint.
  - Unwind GQS once for each class.

- Can achieve exponential reduction in size of state space.
Subformula Tracking

Unwind the GQS w.r.t. non atomic state subformulas

To check $f$

• Choose a good maximal independent set $R = \{R_1, R_2, \ldots, R_k\}$ of state subformulas of $f$

• Replace each $R_i$ in $f$ by a new atomic proposition $r_i$ and obtain $\overline{f}$

• Unwind GQS w.r.t. $R$ to obtain GQS-struct
  • recursively determine which states of the GQS satisfy the subformulas in $R$
  • label a node in GQS-struct with $r_i$ if the corresponding GQS node satisfies $R_i$ (for $i = 1, \ldots, k$)

• Check if $\overline{f}$ is satisfied in GQS-struct
Example

Consider a Resource Controller where process 0 is the controller and others are user processes (i.e. processes 1, 2, … n)

\[ f = E(P_1 \cup g) \]

where \[ g = \bigwedge_{2 \leq i \leq n} E(h(i)) \]

\[ R = \{ P_1, g \} \]

Unwind GQS w.r.t. \( R \) to obtain GQS-struct

Recursively determine states that satisfy \( g \) (use formula decomposition)

Check \( f \) in GQS-struct
In general, subformula tracking and formula decomposition are used recursively.

Good independent sets have symmetric or partially symmetric subformulas.
Implementation

- Extended SMC to PSMC
  priorities can be specified in PSMC
- Used GQS together with
  formula decomposition and
  sub-formula tracking
- Checked *Fire-Wire* Protocol with priorities.
Conclusions.

- Symmetry reductions help in tackling the state explosion.
- Can also be used for systems with less symmetry.
- Have been implemented and used for verifying real-life protocols.

**Future Work.**

- Need to effectively combine with other methods.
- Applications to software verification need to be explored.

Further Details @Home Page:

[www.cs.uic.edu/~sistla](http://www.cs.uic.edu/~sistla)