D-Finder: A Tool for Compositional Deadlock Detection and Verification[1]

DCS Day - Autrans, France

Saddek Bensalem, Marius Bozga, Thanh-Hung Nguyen, Joseph Sifakis

Verimag Laboratory - UJF/CNRS

26 March 2008

[1] CAV 2009 - Grenoble, France

Outline

1. Motivation
2. Compositional verification method
3. Tool Structure
4. Experimentation
5. Conclusions and future work

Verification for concurrent systems

$$B_1 \parallel B_2 \models P \ ?$$

- monolithic verification is hard due to state explosion
- reduced by compositional verification. For example:

$$\frac{B_1 \models \Phi_1,\ B_2 \models \Phi_2,\ C(\Phi_1, \Phi_2, P)}{B_1 \| B_2 \models P}$$

Compositional verification approaches

Assume-guarantee

$$\frac{\begin{array}{c}\langle true\rangle\ B_1\ \langle A\rangle \\ \langle A\rangle\ B_2\ \langle P\rangle\end{array}}{\langle true\rangle B_1 \| B_2\ \langle P\rangle}$$

difficulties [Cobleigh et al., 2008]:

- finding adequate assumptions
- decomposition into sub-systems in case of many components

Invariant basic rule

$$\frac{\begin{array}{c} init \Rightarrow P \\ P\{\tau\}P \quad \forall \tau \in S\end{array}}{S \models \Box P}$$

difficulty: P is an invariant but not inductive

Compositional verification approaches

Invariant general rule

$$\frac{\begin{array}{c} init \Rightarrow Q \\ Q\{\tau\}Q \quad \forall \tau \in S \\ Q \Rightarrow P \end{array}}{S \models \Box P}$$

difficulty: how to compute Q?

An instance of invariant rule

$$\frac{Reach(S) \Rightarrow P}{S \models \Box P}$$

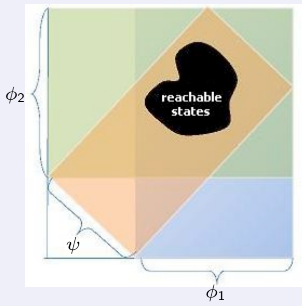
difficulty: computing a set of reachable states $Reach(S)$

D-Finder approach to compositional verification

$$\frac{Reach(S) \subseteq Reach_{App}(S) \quad Reach_{App}(S) \Rightarrow P}{S \models \Box P}$$

Our approach for compositional verification of safety properties (invariants) is based on the following rule:

$$\frac{B_1 \models \Box\Phi_1, B_2 \models \Box\Phi_2,\ \Psi,\ \Phi_1 \wedge \Phi_2 \wedge \Psi \Rightarrow P}{B_1 \parallel B_2 \models \Box P}$$



Outline

The Method: The main Idea

The Method

Compositional verification rule

$$\frac{B_1 \models \Box\Phi_1, B_2 \models \Box\Phi_2, \Psi,\ \ \Phi_1 \wedge \Phi_2 \wedge \Psi \Rightarrow P}{\gamma(B_1, B_2) \models \Box P}$$

- Φ_i is the component invariant of B_i
- Ψ is an interaction invariant computed from Φ_i and $\gamma(B_1, B_2)$
- $\Phi_1 \wedge \Phi_2 \wedge \Psi$ is an over-approximation of reachable states of system

Outline

Automatic Generation Of Component Invariants

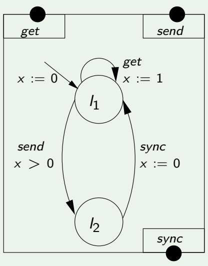
$$\frac{B_1 \models \Box\Phi_1, B_2 \models \Box\Phi_2, \Psi,\ \Phi_1 \wedge \Phi_2 \wedge \Psi \Rightarrow P}{\gamma(B_1, B_2) \models \Box P}$$

Component Invariants

- are over-approximations of the set of reachable states of atomic components
- are computed by using forward propagation [Bensalem et al., 1996]
- $\phi^0 = true \quad \phi^{i+1} = init \vee post(\phi^i)$

Automatic Generation Of Component Invariants

Example



$\Phi = (at\_l_1 \wedge \Phi_{l_1}) \bigvee (at\_l_2 \wedge \Phi_{l_2})$

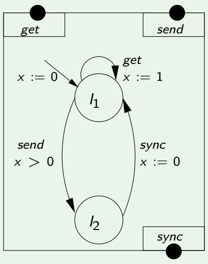
$\Phi_{l_1} = (x = 0) \vee (x = 1)$

$\Phi_{l_2} = (x > 0)$

$\Phi = (at\_l_1 \wedge (x = 1 \vee x = 0)) \bigvee (at\_l_2 \wedge x > 0)$

Automatic Generation Of Component Invariants

Example



$\Phi = (\ at\_l_1 \wedge \Phi_{l_1}) \ \bigvee \ (at\_l_2 \wedge \Phi_{l_2})$

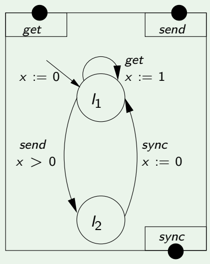
$\Phi_{l_1} = (x = 0) \vee (x = 1)$

$\Phi_{l_2} = (x > 0)$

$\Phi = (\ at\_l_1 \wedge (x = 1 \vee x = 0)) \ \bigvee \ (at\_l_2 \wedge x > 0)$

Automatic Generation Of Component Invariants

Example



$$\Phi = (\, at\_l_1 \wedge \Phi_{l_1}) \bigvee (at\_l_2 \wedge \Phi_{l_2})$$

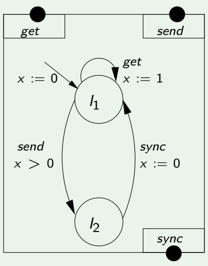
$$\Phi_{l_1} = (x = 0) \vee (x = 1)$$

$$\Phi_{l_2} = (x > 0)$$

$$\Phi = (\, at\_l_1 \wedge (x = 1 \vee x = 0)) \bigvee (at\_l_2 \wedge x > 0)$$

Automatic Generation Of Component Invariants

Example



$$\Phi = (at\_l_1 \wedge \Phi_{l_1}) \bigvee (at\_l_2 \wedge \Phi_{l_2})$$

$$\Phi_{l_1} = (x = 0) \vee (x = 1)$$

$$\Phi_{l_2} = (x > 0)$$

$$\Phi = (at\_l_1 \wedge (x = 1 \vee x = 0)) \bigvee (at\_l_2 \wedge x > 0)$$

Outline

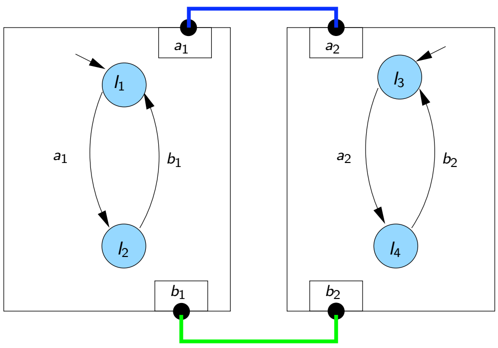
Automatic Generation Of Interaction Invariants

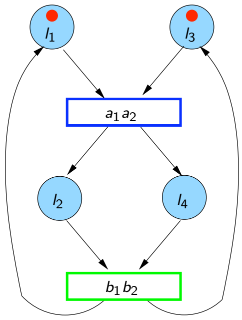
$$\frac{B_1 \models \Box\Phi_1, B_2 \models \Box\Phi_2, \Psi, \ \Phi_1 \wedge \Phi_2 \wedge \Psi \Rightarrow P}{\gamma(B_1, B_2) \models \Box P}$$

Interaction Invariants

- characterize constraints on the global state space induced by synchronizations between components.
- are based on the notion of traps in Petri net.

Computing interaction invariants of systems without data





$\Psi_1 = at\_l_1 \vee at\_l_4$
$\Psi_2 = at\_l_2 \vee at\_l_3$

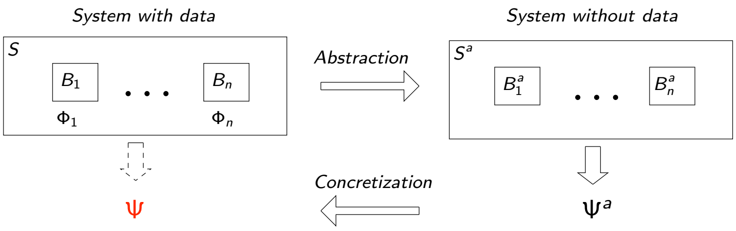
$T_1 = \{l_1, l_4\}$
$T_2 = \{l_2, l_3\}$

Interaction invariant

A trap initially containing a token corresponds to an interaction invariant

Outline

Computing Interaction Invariants of systems with data



Main Idea

Given $\gamma(B_1, \ldots, B_n)$ and a set of component invariants $\Phi_1 \ldots \Phi_n$:

1. Compute an abstract component (without data) B_i^a from B_i and Φ_i
2. Compute interaction invariants Ψ^a for abstract system $\gamma(B_1^a, \ldots, B_n^a)$.
3. Compute concrete invariant Ψ by concretizing Ψ^a

Outline

Checking Invariant Properties and Deadlock-Freedom

Checking Invariant Property Φ

To prove invariance of Φ: find invariants Φ_i, Ψ such that $\bigwedge \Phi_i \wedge \Psi \Rightarrow \Phi$

or equivalently: $\bigwedge \Phi_i \wedge \Psi \wedge \neg\Phi = \mathit{false}$

Checking Deadlock-Freedom

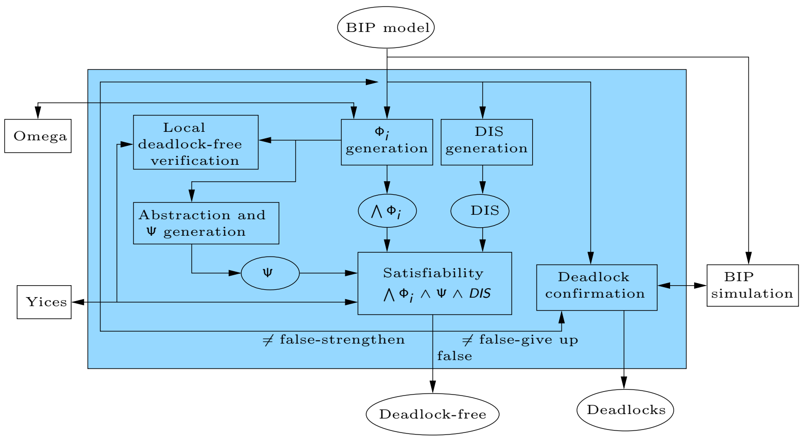
Is a particular case of proving invariants:

- compute DIS - the set of states from which all interactions are disabled
- proving invariance of the predicate $\neg DIS$

Outline

1. Motivation
2. Compositional verification method
   - Component invariants
   - Interaction invariants
   - Abstraction
   - Checking Invariant Properties and Deadlock-Freedom
3. Tool Structure
4. Experimentation
5. Conclusions and future work

D-Finder[2]



[2] http://www-verimag.imag.fr/∼thnguyen/tool/

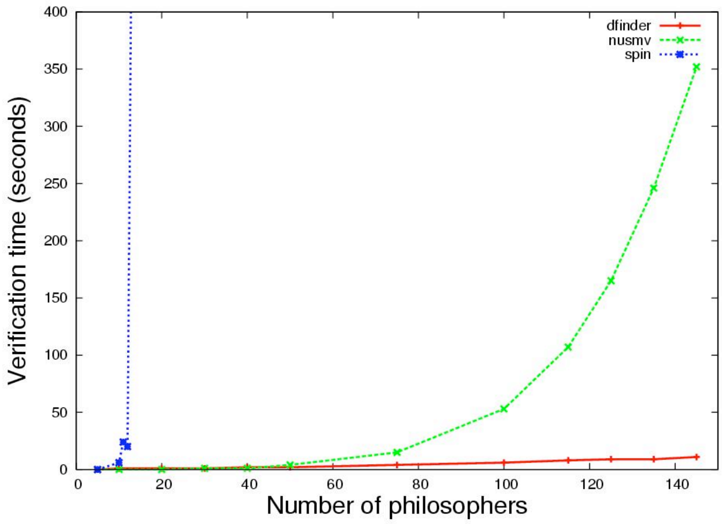
Outline

Case Studies

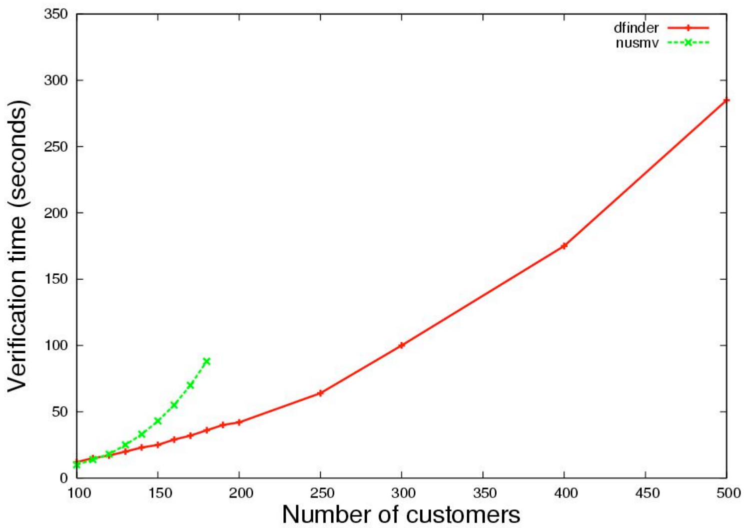
example	n	q	x_b	x_i	$D_{\Phi\Psi}$	t
Philo (10000 Philos)	20000	50000	0	0	3	29m30s
Philo (13000 Philos)	26000	65000	0	0	3	38m48s
Gas station (500 Pums, 5000 Ctms)	5501	20152	0	0	0	18m55s
Readers-Writer(10000 Readers)	10002	20006	0	1	0	36m06s
Smokers (5000 Smokers)	5001	10007	0	0	0	14m
UTS(40 Cars, 256 UCal)	297	795	40	242	0	3m46s
UTS(60 Cars, 625 UCal)	686	1673	60	362	0	25m29s

n number of BIP components in example
q total number of control locations
x_b total number of boolean variables
x_i total number of integer variables
$D_{\Phi\Psi}$ number of potential deadlock configurations remaining in $\bigwedge \Phi_i \wedge \Psi \wedge DIS$
t verification time

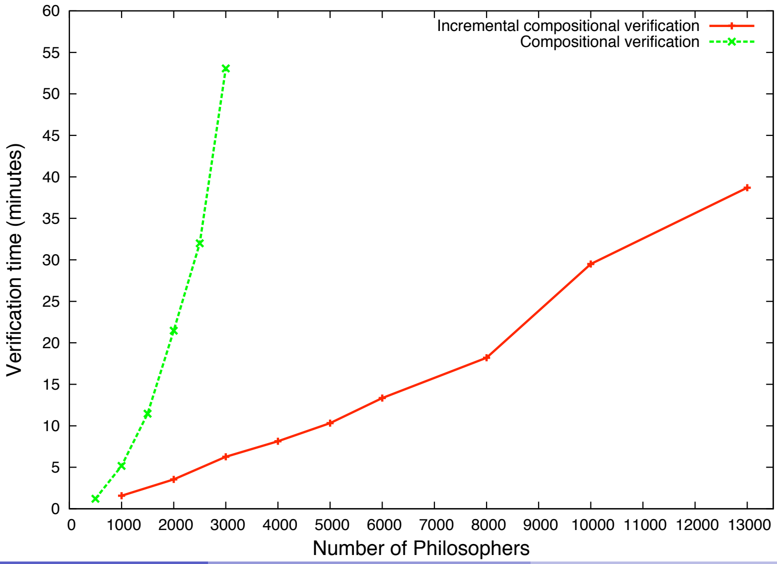
Philosophers - Comparison with NewSMV and SPIN



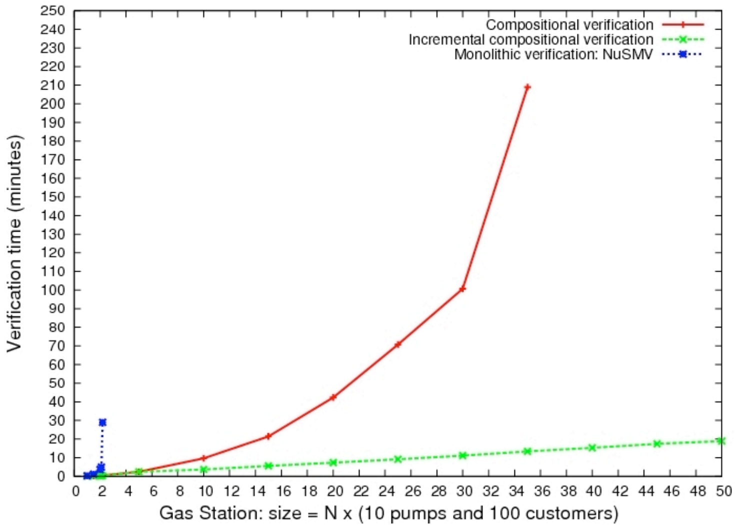
Gas station - Comparison with NewSMV and SPIN



Philosophers - Former and New Method



Gas station - Former and New Method



Outline

Conclusions and future work

Conclusions

- Innovation: using interaction invariant to characterize contexts of individual components.
- Efficiently combines two types of invariants (invariants of atomic components and interaction invariants).
- Using only lightweight analysis techniques

Current and future work

- Adapt to interactions with data transfer
- Strengthen invariants to eliminate potential deadlocks [Bradley and Manna, 2007]

Thank you!