Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel

Fabrice.Kordon@lip6.fr

Réseaux de capteurs...

- Avant tout un système réparti
  - Communications filaires ou non
- Des contraintes embarquées
  - Empreinte mémoire, consommation
- Autres contraintes
  - Mobilité, Temps-réel, fiabilité, variabilité du milieu...
- Dimensions variables
  - Quelques cm... quelques mm... bientôt moins?

Réseaux de capteurs...

- Avant tout un système réparti
  - Communications filaires ou non
- Des contraintes embarquées
  - Empreinte mémoire, consommation
- Autres contraintes
  - Mobilité, Temps-réel, fiabilité, variabilité du milieu...
- Dimensions variables
  - Quelques cm... quelques mm... bientôt moins?
- Besoins en certification
  - Analyse des capacités du système à effectuer ses missions
  - Systèmes complexes = éviter les comportement inattendus

Réseaux de capteurs...

F. Kordon — LIP6/MoVe — UPMC 2 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

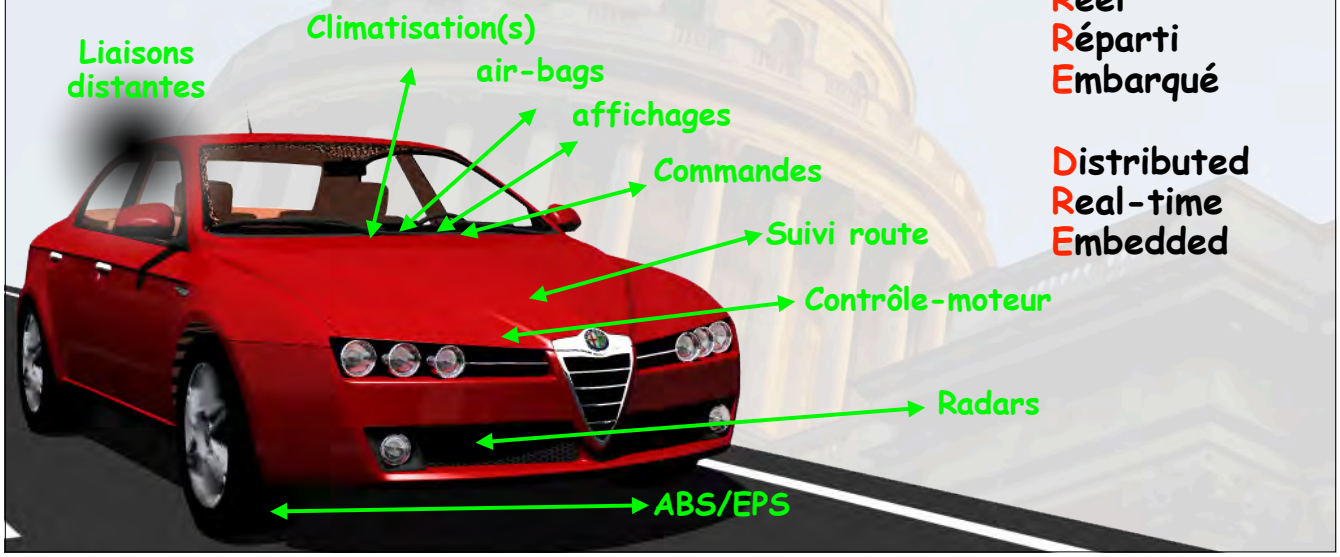
- Avant tout un système réparti
  - Communications filaires ou non
- Des contraintes embarquées
  - Empreinte mémoire, consommation
- Autres contraintes
  - Mobilité, Temps-réel, fiabilité, variabilité du milieu...
- Dimensions variables
  - Quelques cm... quelques mm... bientôt moins?
- Besoins en certification
  - Analyse des capacités du système à effectuer ses missions
  - Systèmes complexes = éviter les comportement inattendus

Systèmes embarqués aujourd'hui, un exemple

- Un véhicule = un système réparti embarqué
  - Fonctions locales nombreuses : AUTOSAR = middleware
  - Système critique => Fiabilité
  - Liaisons distantes (IVI = In-Vehicle Infotainment)

Temps
Réel
Réparti
Embarqué

Distributed
Real-time
Embedded

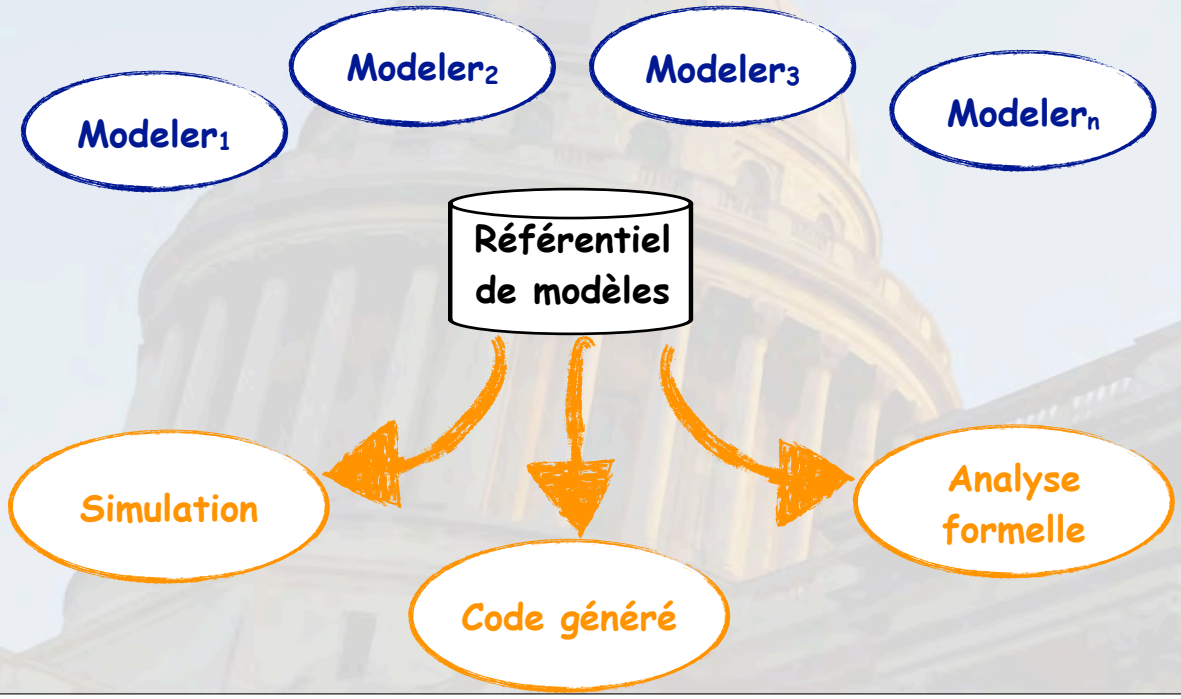


Vision actuelle (et idéale) du Model Driven Engineering?

- Automatiser = prévenir les incertitudes du développement

Vision actuelle (et idéale) du Model Driven Engineering?

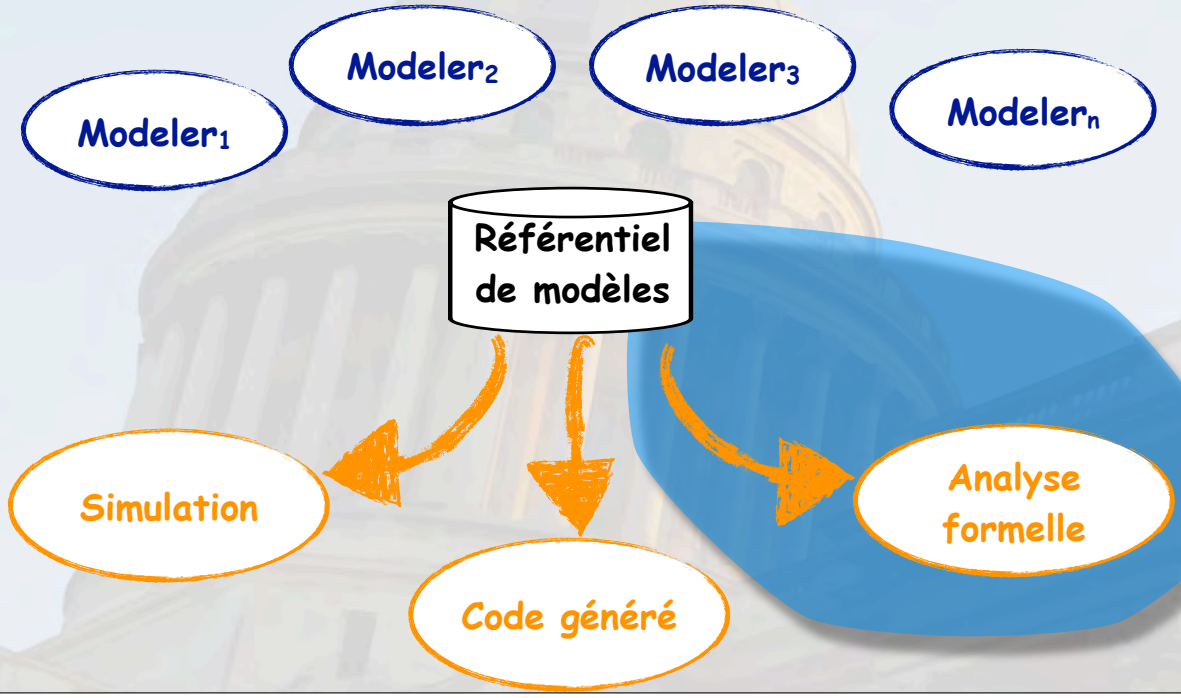
Automatiser = prévenir les incertitudes du développement



Vision actuelle (et idéale) du Model Driven Engineering?

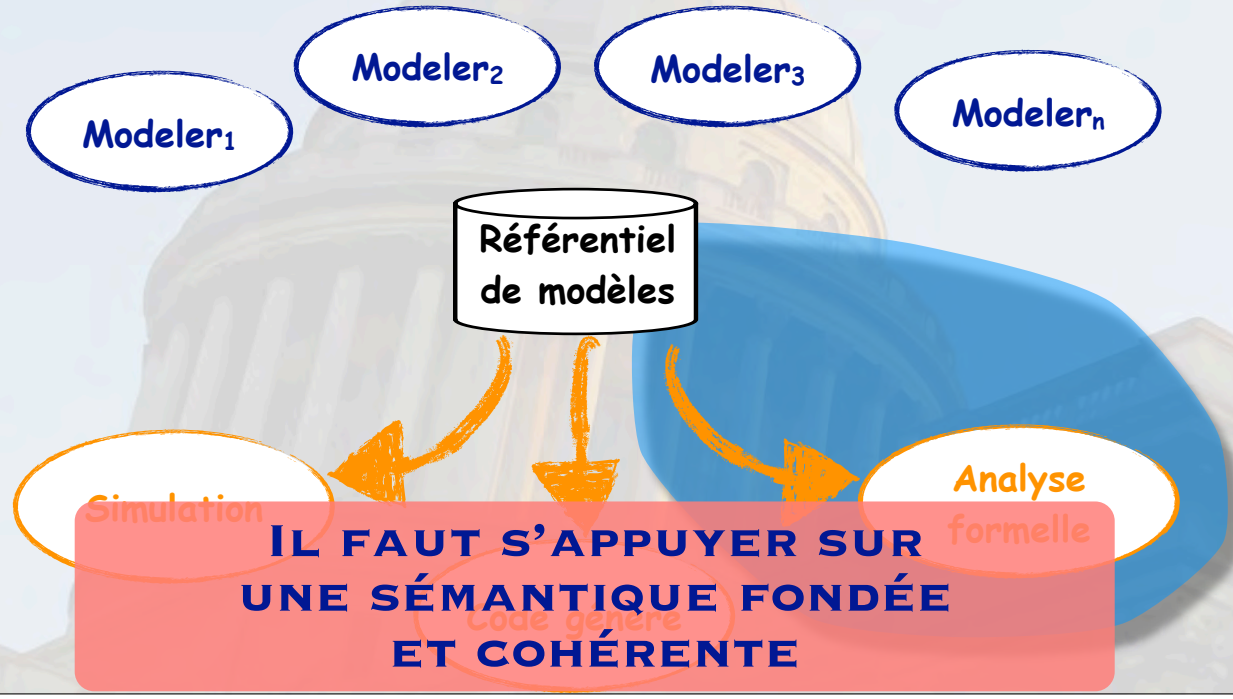
F. Kordon — LIP6/MoVe — UPMC 4 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

Automatiser = prévenir les incertitudes du développement



Vision actuelle (et idéale) du Model Driven Engineering?

Automatiser = prévenir les incertitudes du développement

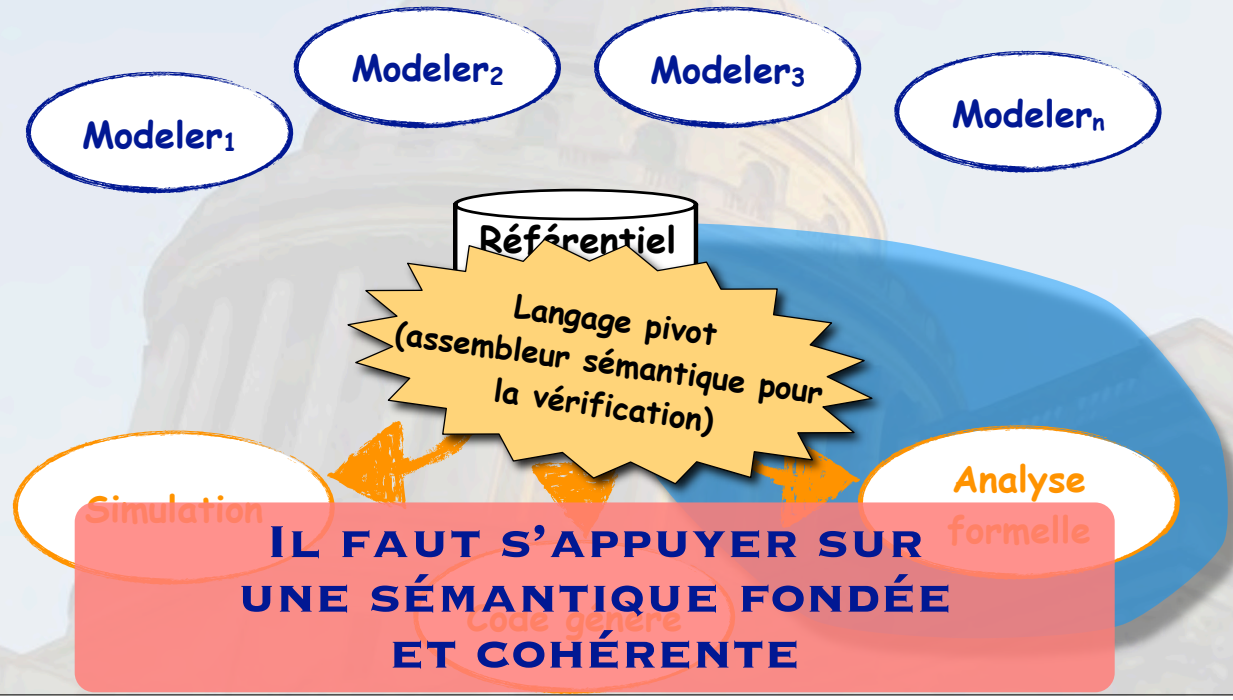


Il faut s'appuyer sur une sémantique fondée et cohérente

Vision actuelle (et idéale) du Model Driven Engineering?

F. Kordon — LIP6/MoVe — UPMC 4 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

Automatiser = prévenir les incertitudes du développement



Il faut s'appuyer sur une sémantique fondée et cohérente

Analyse formelle?

- Preuve formelle (approche «algébrique»)
  - Paramétré
  - Peu automatisé
  - Diagnostic difficile
- Analyse Statique
  - Approche statique (structure plus simple)
  - Capture du comportement délicate (vision structurelle)
- Model Checking
  - Automatisable
  - Diagnostic aisé
  - Explosion combinatoire

Analyse formelle?

F. Kordon — LIP6/MoVe — UPMC 5 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

- Preuve formelle (approche «algébrique»)
  - Paramétré
  - Peu automatisé
  - Diagnostic difficile
- Analyse Statique
  - Approche statique (structure plus simple)
  - Capture du comportement délicate (vision structurelle)
- Model Checking
  - Automatisable
  - Diagnostic aisé
  - Explosion combinatoire

Analyse formelle?

- Preuve formelle (approche «algébrique»)
  - Paramétré
  - Peu automatisé
  - Diagnostic difficile
- Analyse Statique
  - Approche statique (structure plus simple)
  - Capture du comportement délicate (vision structurelle)
- Model Checking
  - Automatisable
  - Diagnostic aisé
  - Explosion combinatoire

Sujet de cet exposé

Approches complémentaires

Analyse formelle?

F. Kordon — LIP6/MoVe — UPMC 5 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

- Preuve formelle (approche «algébrique»)
  - Paramétré
  - Peu automatisé
  - Diagnostic difficile
- Analyse Statique
  - Approche statique (struc
  - Capture du comportemen structurelle)
- Model Checking
  - Automatisable
  - Diagnostic aisé
  - Explosion combinatoire

Sujet de cet exposé

Ennemis du model checking:
1) Consommation mémoire
2) consommation CPU

Model Checking: éléments «à charge» et «à décharge»

TR^2E = systèmes complexes

- Interoperabilité
  - Communications asynchrones => Complexité

- Dynamicité
  - Pas forcément de borne => encore plus de complexité

- Symétries
  - Schéma de répétition dans le système
  - Permutabilité de certains éléments

- Structuration
  - Traiter la localité dans certains systèmes
  - Partager des portions communes des configurations

- Hiérarchie
  - Meilleure organisation et gestion des «répétitions»

Et maintenant?

- Quelques observations sur ces systèmes complexes
  - Exploiter les «bonnes» caractéristiques
- Une proposition: «Instantiable Transition Systems»
  - Sans temps
  - Encodage du temps
- Liaison à une approche «MDE»
- Éléments de conclusion...

... avant exposé suivant

Et maintenant?

- Quelques observations sur ces systèmes complexes
  - Exploiter les «bonnes» caractéristiques
- Une proposition: «Instantiable Transition Systems»
  - Sans temps
  - Encodage du temps
- Liaison à une approche «MDE»
- Éléments de conclusi… …t exposé suivant

Observations

Hierarchy is good for distributed systems

[Hong 2012]

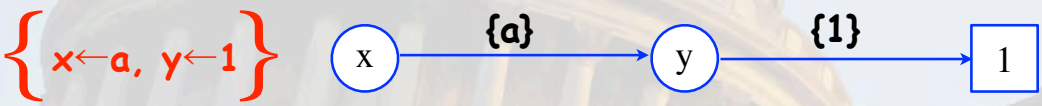
Symbolic Encoding of the State Space with Decision Diagrams

- Based on the notion of locality [Bryant 1986] + [Clarke 1992]
  - State = vector of integers
  - Share of common parts in a set of states
- Sequence of variable affectations
  - Accepting sequence: x←a, y←1 and x←a, y←2
- Exponential gain in favorable cases
  - Order of the variable encoding is crucial

Symbolic Encoding of the State Space with Decision Diagrams

F. Kordon — LIP6/MoVe — UPMC 9 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

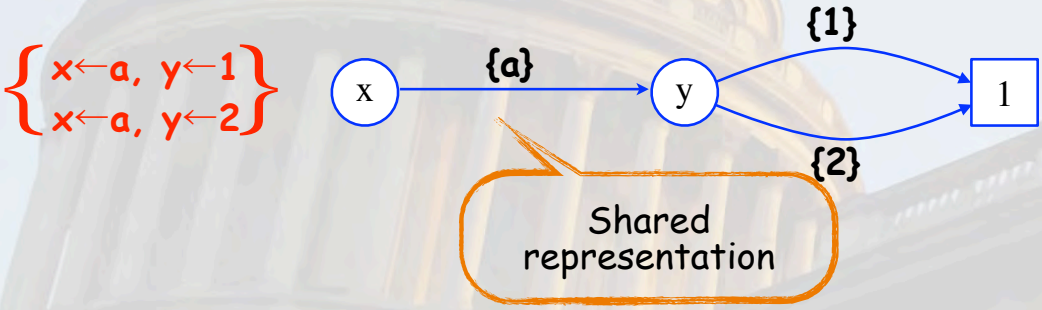
- Based on the notion of locality [Bryant 1986] + [Clarke 1992]
  - State = vector of integers
  - Share of common parts in a set of states
- Sequence of variable affectations
  - Accepting sequence: x←a, y←1 and x←a, y←2



- Exponential gain in favorable cases
  - Order of the variable encoding is crucial

Symbolic Encoding of the State Space with Decision Diagrams

- Based on the notion of locality [Bryant 1986] + [Clarke 1992]
  - State = vector of integers
  - Share of common parts in a set of states
- Sequence of variable affectations
  - Accepting sequence: x←a, y←1 and x←a, y←2

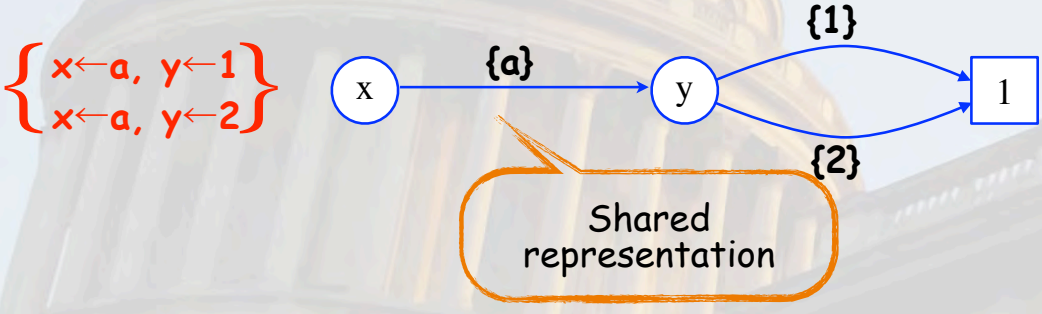


- Exponential gain in favorable cases
  - Order of the variable encoding is crucial

Symbolic Encoding of the State Space with Decision Diagrams

F. Kordon — LIP6/MoVe — UPMC 9 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

Based on the notion of locality [Bryant 1986] + [Clarke 1992]
State = vector of integers
Share of common parts in a set of states
Sequence of variable affectations
Accepting sequence: x←a, y←1 and x←a, y←2



Exponential gain in favorable cases
Order of the variable encoding is crucial

More required to have hierarchy

- Hierarchical Decision Diagrams [Couvreur 2005]
  - Arcs labeled by sets
  - A decision diagram represents a set
  - Recursivity
- Example
  - Structured data: <p1,{1,3}> + <p2,{1,3}>

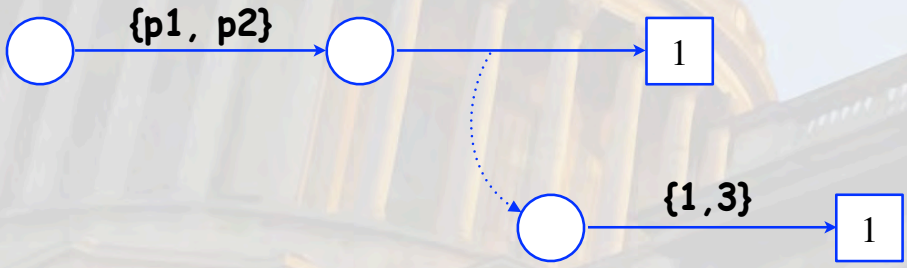
More required to have hierarchy

F. Kordon — LIP6/MoVe — UPMC 10 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

- Hierarchical Decision Diagrams [Couvreur 2005]
  - Arcs labeled by sets
  - A decision diagram represents a set
  - Recursivity
- Example
  - Structured data: <p1,{1,3}> + <p2,{1,3}>

More required to have hierarchy

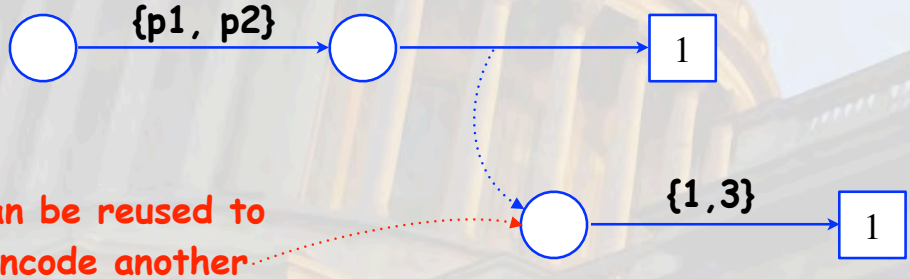
- Hierarchical Decision Diagrams [Couvreur 2005]
  - Arcs labeled by sets
  - A decision diagram represents a set
  - Recursivity
- Example
  - Structured data: <p1,{1,3}> + <p2,{1,3}>



More required to have hierarchy

F. Kordon — LIP6/MoVe — UPMC 10 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

- Hierarchical Decision Diagrams [Couvreur 2005]
  - Arcs labeled by sets
  - A decision diagram represents a set
  - Recursivity
- Example
  - Structured data: <p1,{1,3}> + <p2,{1,3}>

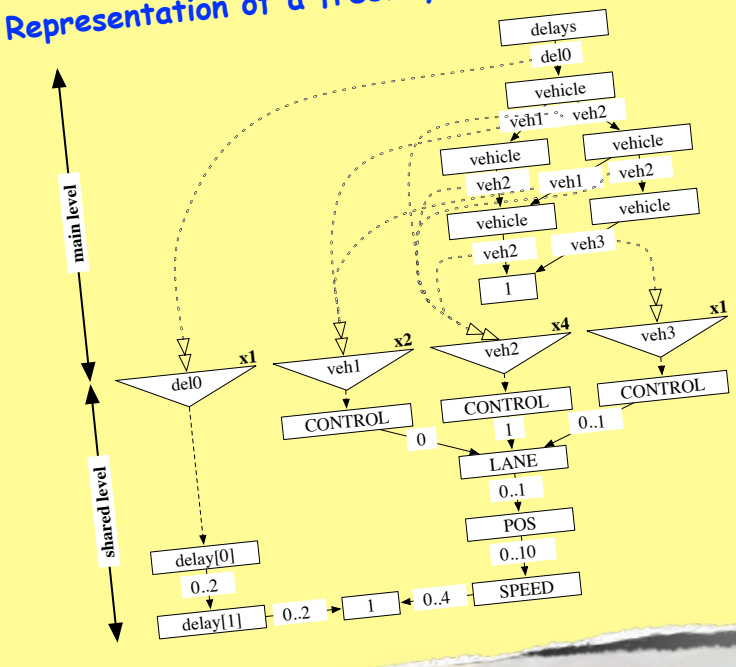


Can be reused to encode another part of the system

More required to have hierarchy

Hierarchical ... vreur 2005]

Representation of a freeway [Bérard 2008]



1

,3}

1

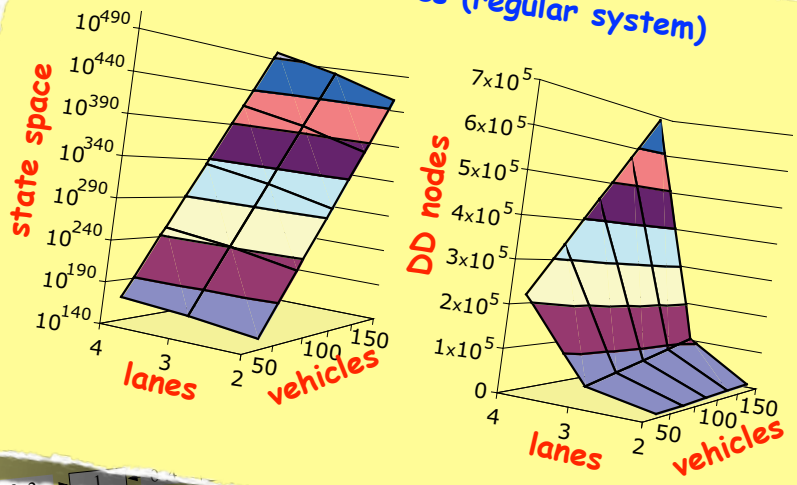
More required to have hierarchy

F. Kordon — LIP6/MoVe — UPMC 10 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

Hierarchical ... vreur 2005]

Representation of a freeway [Bérard 2008]

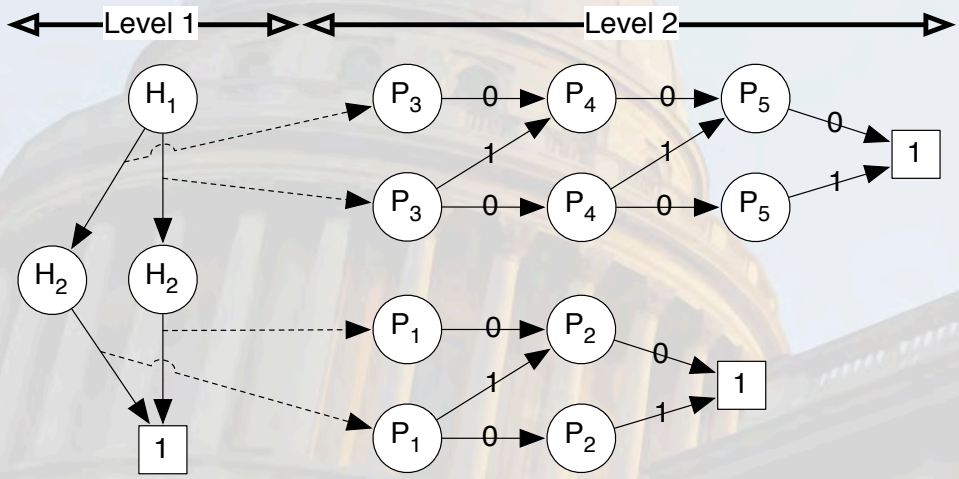
Performances (regular system)



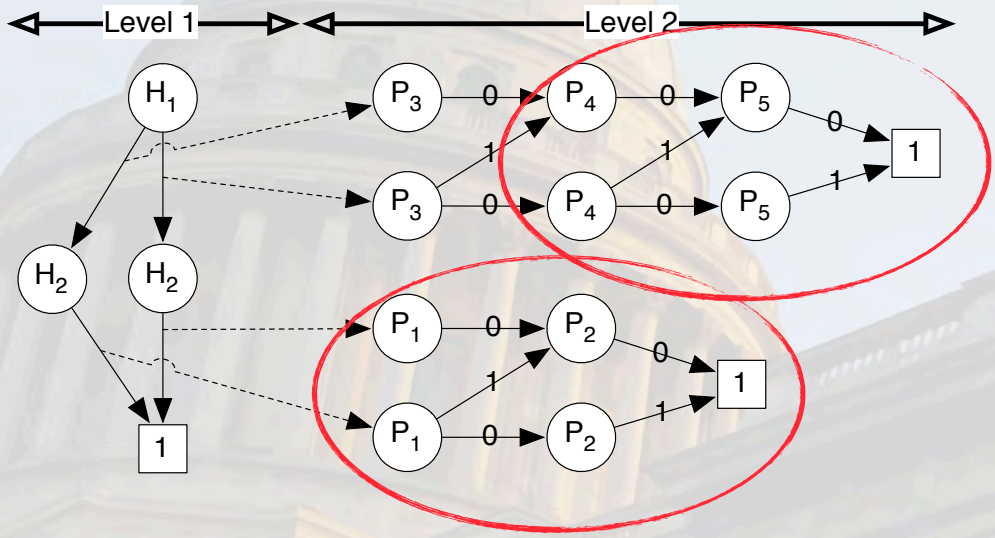
PNXDD, A FIRST APPLICATION

- Objective: exploit the structure of large Petri Nets
  - Obtained from «unfolding of Colored Nets»
  - Characteristics: large models with repeated patterns
- Involved techniques
  - Compute a variable order
    - FORCE [Aloul 2003] or NOA99 [Heiner 2009]
  - Hierarchically cluster this order
- Anonymize the clusters as much as possible
  - Anonymization = contextual interpretation
  - Reused patterns contextually

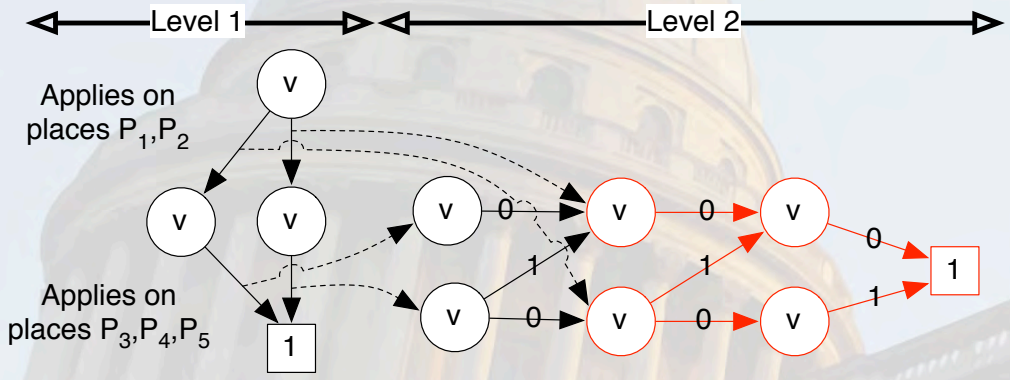
ANONYMIZATION...



Anonymization...

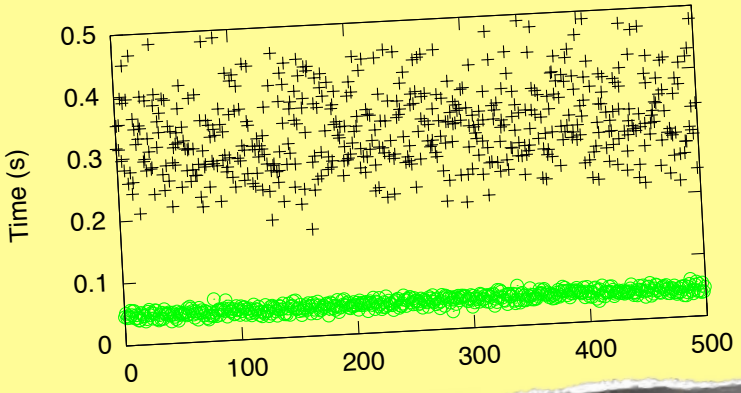


Anonymization...



PNXDD: PERFORMANCES

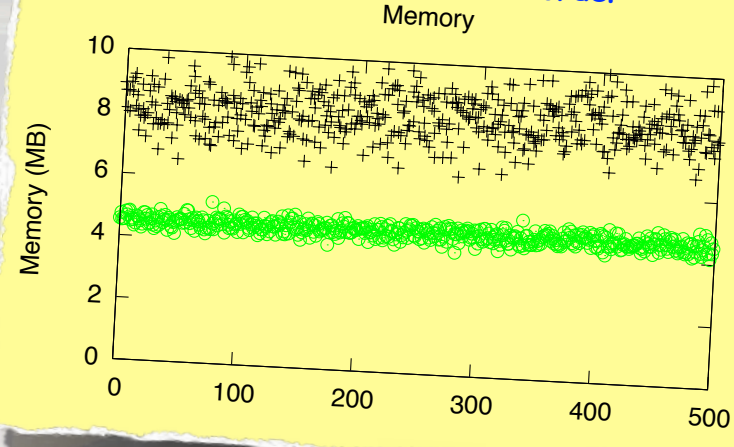
Comparing to random order



PNXDD: PERFORMANCES

F. Kordon — LIP6/MoVe — UPMC 13 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

Comparing to random order



PNXDD: PERFORMANCES

PolyORB case Study (1 state = up to 2KB)

number of Instances	State Space Size	Flat	Flat order performance				Hierarchical order performance				Gain (in %)			
			Final	Peak	Time	MB	Final	Peak	Time	MB	Final	Peak	Time	Mem
2	1.6×10^6	F	223,243	3.1×10^6	580.8	1,316	27,548	491,803	29.3	352	88	84	95	73
		N	78,785	451,494	98.7	273	10,050	127,791	6.2	81	87	72	94	70
3	2.8×10^7	F	593,363	1.2×10^7	4,708	2,851	78,067	2.1×10^6	188.7	692	87	83	96	76
		N	280,068	2.5×10^6	948.8	1,513	33,526	524,288	64.3	358	88	79	93	76
4	2.1×10^8	F	1.2×10^6	3.1×10^7	8,310	5,765	146,589	4.2×10^6	528.7	1,780	87	86	94	69
		N	666,886	8.4×10^6	217,173	2,457	84,126	2.1×10^6	326.0	1,451	87	75	99	41
5	1.4×10^9	F	TOF	TOF	TOF	TOF	143,903	1.1×10^7	2,361.6	3,045	∞	∞	∞	∞
		N	TOF	TOF	TOF	TOF	87,875	4.2×10^6	1,045.1	2,216	∞	∞	∞	∞
6	9.2×10^9	F	TOF	TOF	TOF	TOF	288,649	2.2×10^7	15,757	6,144	∞	∞	∞	∞
		N	TOF	TOF	TOF	TOF	140,565	1.5×10^7	19,474	4,865	∞	∞	∞	∞
7	-	F	TOF	TOF	TOF	TOF	MOF	MOF	MOF	MOF	-	-	-	-
		N	TOF	TOF	TOF	TOF	TOF	TOF	TOF	TOF	-	-	-	-

PNXDD: PERFORMANCES

PolyORB case Study (1 state = up to 2KB)

number of Instances	State Space Size	Flat	Flat order performance				Hierarchical order performance				Gain (in %)			
			Final	Peak	Time	MB	Final	Peak	Time	MB	Final	Peak	Time	Mem
2	1.6×10^6	F	223,243	3.1×10^6	580.8	1,316	27,548	491,803	29.3	352	88	84	95	73
		N	78,785	451,494	98.7	273	10,050	127,791	6.2	81	87	72	94	70
3	2.8×10^7	F	593,363	1.2×10^7	4,708	2,851	78,067	2.1×10^6	188.7	692	87	83	96	76
		N	280,068	2.5×10^6	948.8	1,513	33,526	524,288	64.3	358	88	79	93	76
4	2.1×10^8	F	1.2×10^6	3.1×10^7	8,310	5,765	146,589	4.2×10^6	528.7	1,780	87	86	94	69
		N	666,886	8.4×10^6	217,173	2,457	84,126	2.1×10^6	326.0	1,451	87	75	99	41
5	1.4×10^9	F	TOF	TOF	TOF	TOF	143,903	1.1×10^7	2,361.6	3,045	∞	∞	∞	∞
		N	TOF	TOF	TOF	TOF	87,875	4.2×10^6	1,045.1	2,216	∞	∞	∞	∞
6	9.2×10^9	F	TOF	TOF	TOF	TOF	288,649	2.2×10^7	15,757	6,144	∞	∞	∞	∞
		N	TOF	TOF	TOF	TOF	140,565	1.5×10^7	19,474	4,865	∞	∞	∞	∞
7	-	F	TOF	TOF	TOF	TOF	MOF	MOF	MOF	MOF	-	-	-	-
		N	TOF	TOF	TOF	TOF	TOF	TOF	TOF	TOF	-	-	-	-

PNXDD: PERFORMANCES

NEOPPOD case Study (1 state = up to 2.4KB)

number of Instances	State Space Size	Flat	Flat order performance				Hierarchical order performance				Gain (in %)			
			Final	Peak	Time	MB	Final	Peak	Time	MB	Final	Peak	Time	Mem
2	194	F	202	679	0.024	4.1	80	217	0.007	3.42	60	68	71	17
		N	463	1,688	0.024	4.5	154	558	0.011	3.57	67	67	55	21
3	90,861	F	5,956	48,269	0.86	19.0	1,126	12,491	0.15	8.2	81	74	82	57
		N	3,820	23,974	0.76	12.2	709	4,838	0.10	6.7	81	80	87	45
4	9.7×10^8	F	84,398	1.0×10^6	62.6	338.8	11,728	178,921	5.0	70.83	86	82	92	79
		N	155,759	1.3×10^6	186.1	490.3	19,875	217,536	0.007	114.2	87	83	99	77

PRELIMINARY CONCLUSION

- Combination of techniques can help tackling complexity
  - MCC@Petri Net 2013 (as well as in 2011 and 2012) showed this
  - Complex distributed systems enable «good techniques»
    - Suitable for Systems of Systems
    - Complexity: «additive logic» instead of «product logic»
- But Distributed systems are also Embedded
  - OK for the main characteristics...

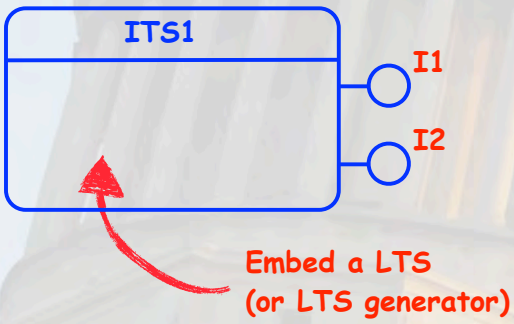
    ... but we need time!
- Need for a formal assembly language to manage time
  - ITS (Instantiable Transition Systems) [Thierry-Mieg 2009]
    - Combine: symmetries + decision diagrams + saturation
    - How to extend for time!

Instantiable Transition Systems and Time

[Thierry-Mieg 2011]

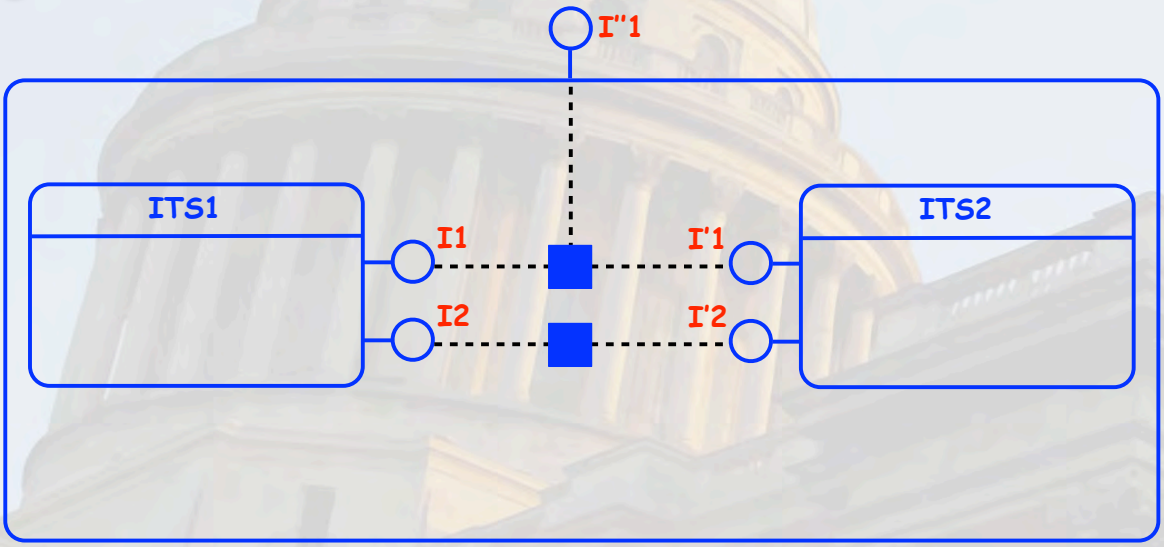
ITS in a Nutshell..

- A formal framework for verification
  - Elementary ITS



ITS in a Nutshell..

- A formal framework for verification
  - Elementary ITS
  - Composite ITS
  - Recursion...

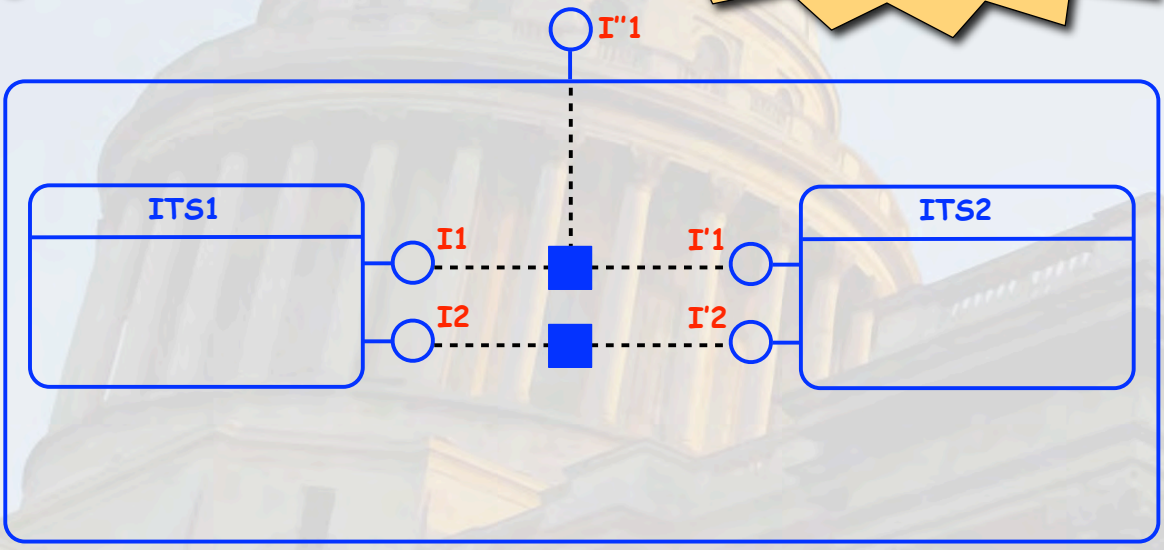


ITS in a Nutshell..

F. Kordon — LIP6/MoVe — UPMC 16 Mo t, Septembre 2013

- A formal framework for
  - Elementary ITS
  - Composite ITS
  - Recursion...

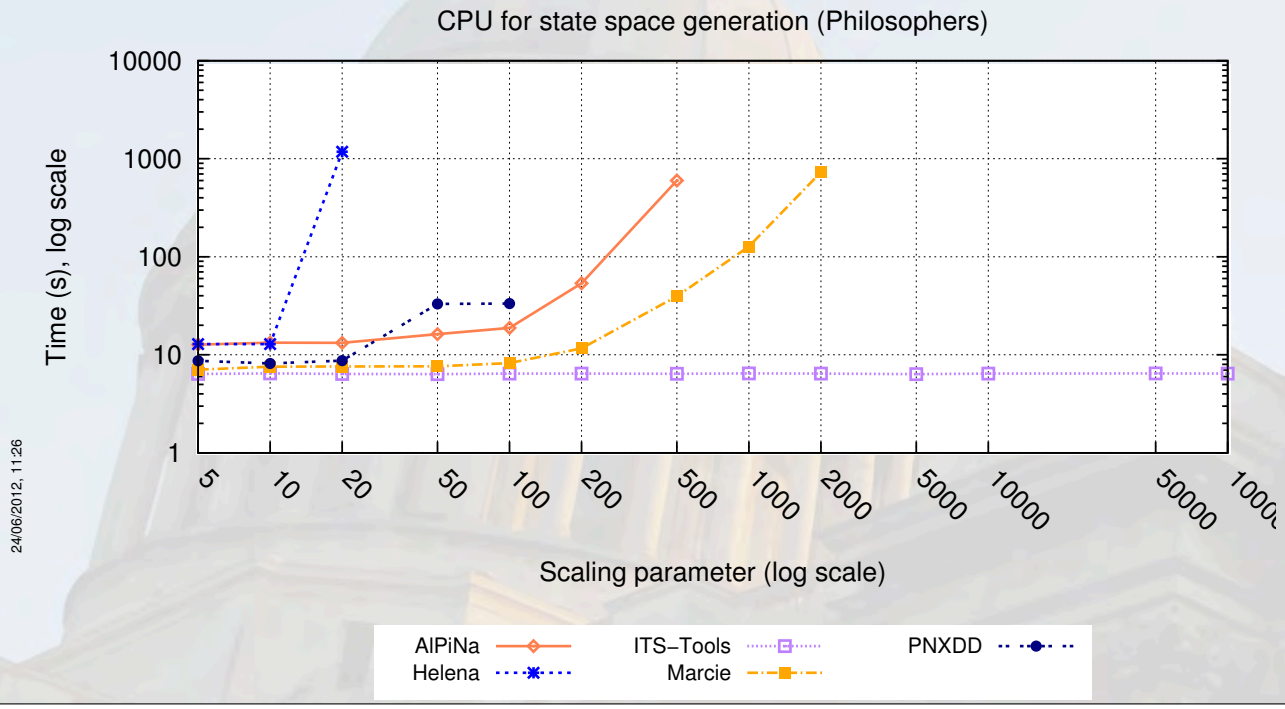
Advantages:
Express locality
Define instances
Hierarchy



The «recursive unfolding» technique

PERFORMANCES (MCC 2012 @ PETRI NETS)
F. Kordon — LIP6/MoVe — UPMC 17 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

The «recursive unfolding» technique

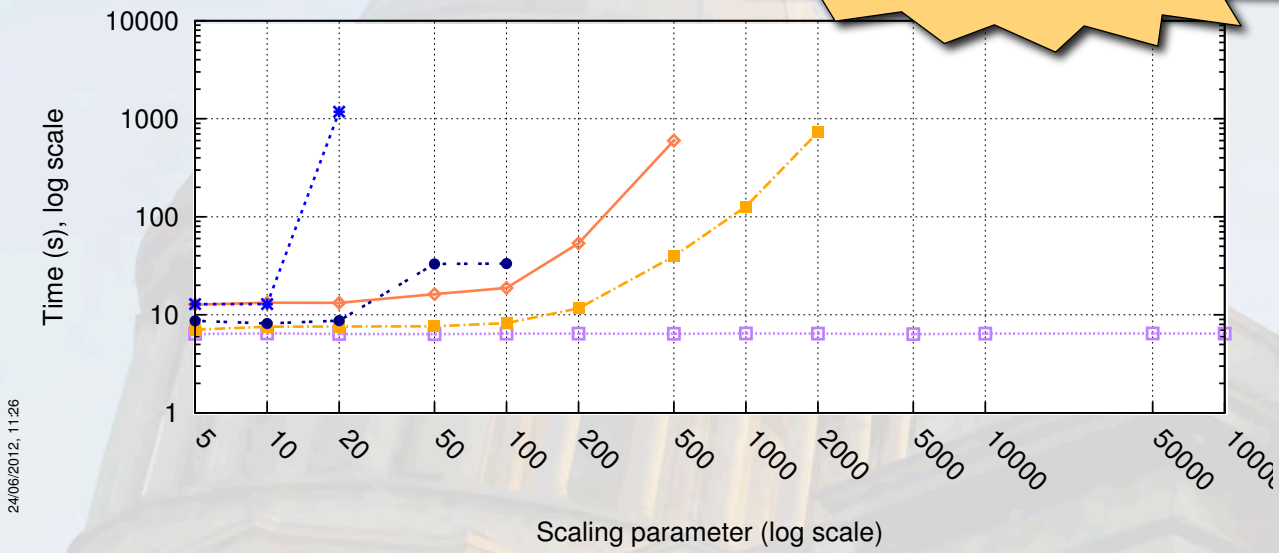


Performances (MCC 2012 @ Petri Nets)

The «recursive unfolding» tec

Memory limit : 4GB
CPU limit: 1h
None reached with ITS

CPU for state space gene



AlPiNa
Helena
ITS–Tools
Marcie
PNXDD

24/06/2012, 11:26

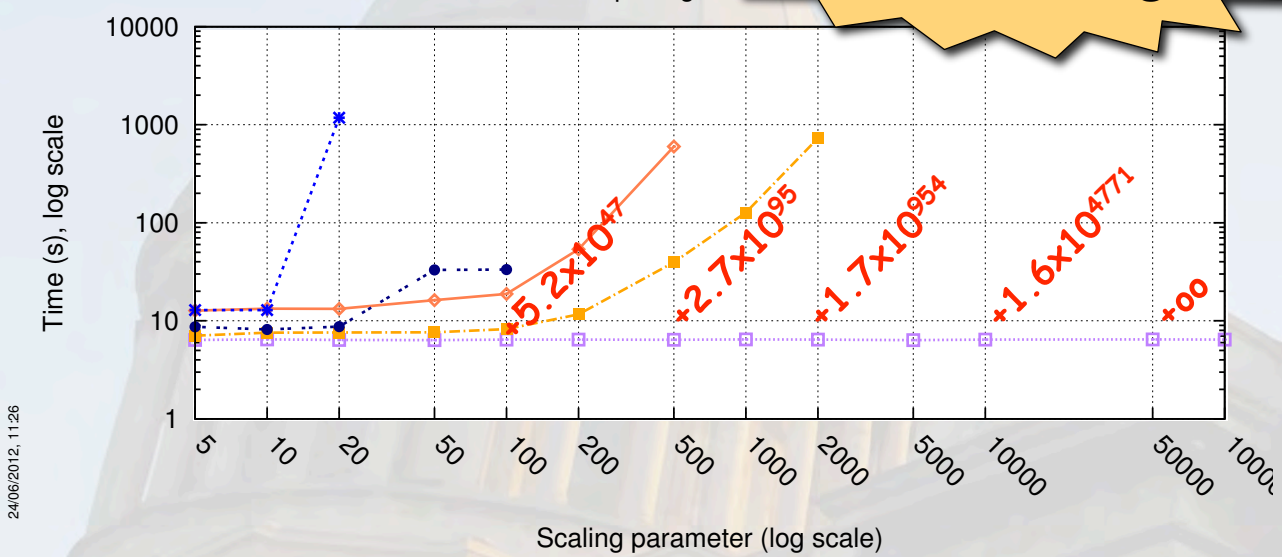
Performances (MCC 2012 @ Petri Nets)

F. Kordon — LIP6/MoVe — UPMC 17 Modèles Hiérarchi tembre 2013

The «recursive unfolding» tec

Memory limit : 4GB
CPU limit: 1h
None reached with ITS

CPU for state space gene



AlPiNa
Helena
ITS–Tools
Marcie
PNXDD

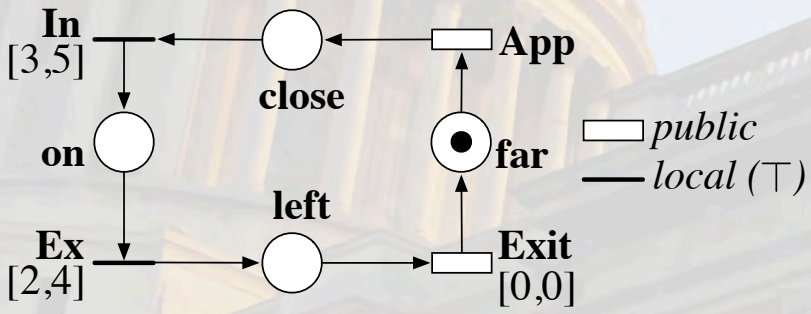
24/06/2012, 11:26

And what about time?

- Discrete time can be encoded in a similar way
  - An extra interface : $\mathbb{1}$
  - All $\mathbb{1}$ are synchronized
  - Caution: consistency of synchronized time constraints
    - Only synchronize untimed transitions?
    - Only synchronize transitions with the same time constraints?
- Then, all the underlying tricks can be activated
  - Deduction of locality
  - Various encoding patterns
  - Rewriting mechanisms to enable «saturation»
  - etc...

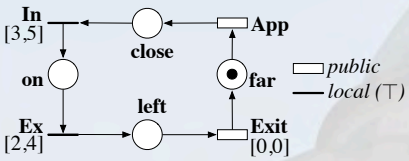
Example: trains crossing...

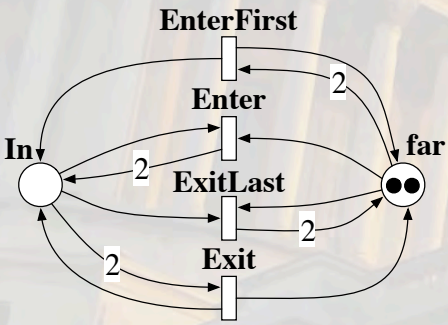
- Here, ITS embed Time Petri Nets



Example: trains crossing...

Here, ITS embed Time Petri Nets

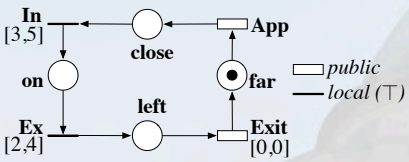


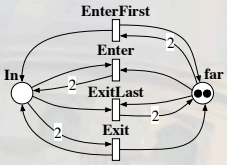


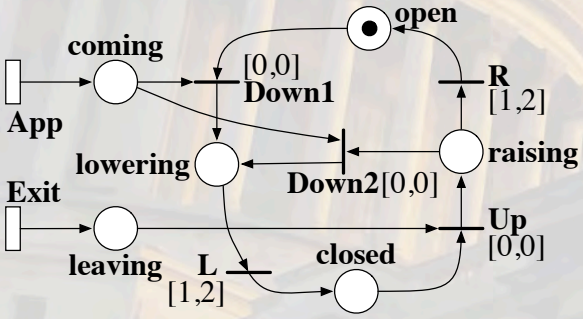
Example: trains crossing...

F. Kordon — LIP6/MoVe — UPMC 19 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

Here, ITS embed Time Petri Nets

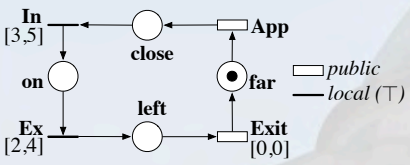


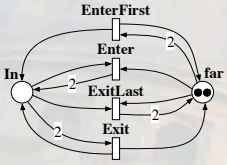


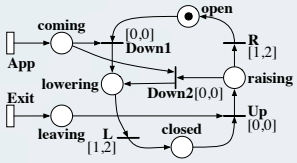


Example: trains crossing...

Here, ITS embed Time Petri Nets





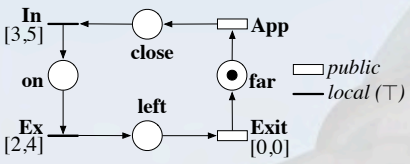


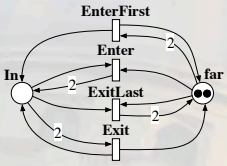
t_0: train	t_1: train	λ
App	ε	App
ε	App	App
Exit	ε	Exit
ε	Exit	Exit
𝟙	𝟙	𝟙

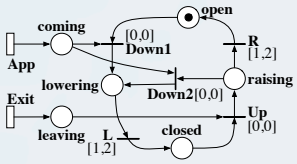
Example: trains crossing...

F. Kordon — LIP6/MoVe — UPMC 19 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

Here, ITS embed Time Petri Nets





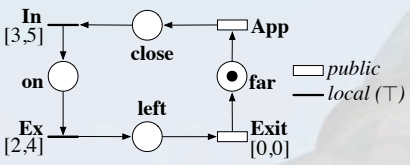


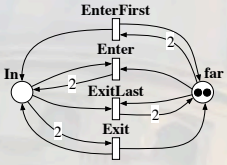
t_0: train	t_1: train	λ
App	ε	App
ε	App	App
Exit	ε	Exit
ε	Exit	Exit
𝟙	𝟙	𝟙

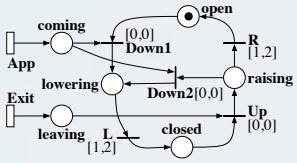
tg: TrainGroup	cc: ContrGate	λ
Exit	Exit	⊤
App	App	⊤
𝟙	𝟙	𝟙

Example: trains crossing...

Here, ITS embed Time Petri Nets







t_0: train	t_1: train	λ
App	ε	App
ε	App	App
Exit	ε	Exit
ε	Exit	Exit
𝟙	𝟙	𝟙

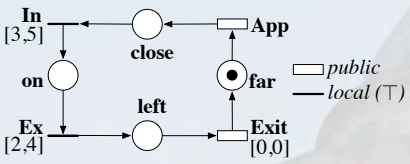
tg: TrainGroup	cc: ContrGate	λ
Exit	Exit	⊤
App	App	⊤
𝟙	𝟙	𝟙

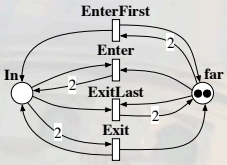
g: gate	c: controller	λ
App	EnterFirst	App
ε	Enter	App
Exit	ExitLast	Exit
ε	Exit	Exit
𝟙	𝟙	𝟙

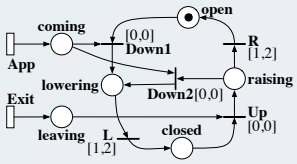
Example: trains crossing...

F. Kordon — LIP6/MoVe — UPMC 19 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

Here, ITS embed Time Petri Nets







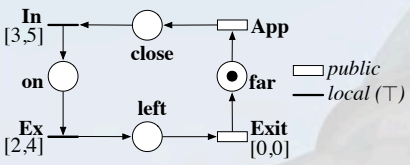
t_0: train	t_1: train	λ
App	ε	App
ε	App	App
Exit	ε	Exit
ε	Exit	Exit
𝟙	𝟙	𝟙

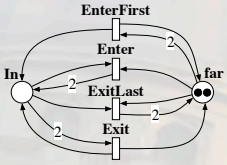
tg: TrainGroup	cc: ContrGate	λ
Exit	Exit	⊤
App	App	⊤
𝟙	𝟙	𝟙

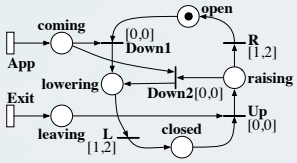
g: gate	c: controller	λ
App	EnterFirst	App
ε	Enter	App
Exit	ExitLast	Exit
ε	Exit	Exit
𝟙	𝟙	𝟙

Example: trains crossing...

Here, ITS embed Time Petri Nets







Global synchronization for time elapse

(may reduce some symmetries)

t_0: train	t_1: train	λ
App	ε	App
ε	App	App
Exit	ε	Exit
ε	Exit	Exit
𝟙	𝟙	𝟙

tg: TrainGroup	cc: ContrGate	λ
Exit	Exit	⊤
App	App	⊤
𝟙	𝟙	𝟙

g: gate	c: controller	λ
App	EnterFirst	App
ε	Enter	App
Exit	ExitLast	Exit
ε	Exit	Exit
𝟙	𝟙	𝟙

Example: trains crossing...

Here, ITS embed Time Petri Nets

Groups of 4 trains?

t_0: train	t_1: train	t_2: train	t_3: train	λ
App	ε	ε	ε	App
ε	App	ε	ε	App
ε	ε	App	ε	App
ε	ε	ε	App	App
Exit	ε	ε	ε	Exit
ε	Exit	ε	ε	Exit
ε	ε	Exit	ε	Exit
ε	ε	ε	Exit	Exit
𝟙	𝟙	𝟙	𝟙	𝟙

Global synchronization for time elapse

(may reduce some symmetries)

g: gate	c: controller	λ
App	EnterFirst	App
ε	Enter	App
Exit	ExitLast	Exit
ε	Exit	Exit
𝟙	𝟙	𝟙

Example: trains crossing...

Here, ITS embed Time Petri N

Recursive unfolding possible
2x(2x(2x(train)))...

Groups of 4 trains?

t_0: train	t_1: train	t_2: train	t_3: train	λ
App	ε	ε	ε	App
ε	App	ε	ε	App
ε	ε	App	ε	App
ε	ε	ε	App	App
Exit	ε	ε	ε	Exit
ε	Exit	ε	ε	Exit
ε	ε	Exit	ε	Exit
ε	ε	ε	Exit	Exit
𝟙	𝟙	𝟙	𝟙	𝟙

for time elapse

ymmetries)

g: gate	c: controller	λ
App	EnterFirst	App
ε	Enter	App
Exit	ExitLast	Exit
ε	Exit	Exit
𝟙	𝟙	𝟙

Example: trains crossing...

F. Kordon — LIP6/MoVe — UPMC 19 Modèles Hiérarchiques pour la vérification efficace de systèmes temps-réel — Rabat, Septembre 2013

Here, ITS embed Time Petri Nets

Integration in Romeo, good performances against known tools

Train (N is the number of trains)

	Roméo			RED		UPPAAL/sym			Roméo/SDD		
N	*tm*	*mm*	*sm*	*tm*	*mm*	*tm*	*mm*	*sm*	*tm*	*mm*	*sm*
6	43.1	36 948	29 640	7	202 412	0.14	908	432	1.5	7 360	$4.83\ 10^6$
7	6 115	377 452	131 517	66	723 428	0.23	3 200	957	2.5	10 304	$6.28\ 10^7$
8	DNF	-	-	-	OOM	1	3 336	2 078	4	14 188	$8.16\ 10^8$
13	-	-	-	-	-	2 634	13 188	79 598	26	56 660	$3.02\ 10^{14}$
15	-	-	-	-	-	60 860	61 256		42	86 360	$5.11\ 10^{16}$
16	-	-	-	-	-	DNF	-	-	52	104 848	$6.65\ 10^{17}$
44	-	-	-	-	-	-	-	-	1143	$2.13\ 10^6$	$1.03\ 10^{49}$

Example: trains crossing...

Here, ITS embed Time Petri Nets

Integration in Romeo, good performances against known tools

Train (N is the number of trains)

	Roméo			RED		UPPAAL/sym			Roméo/SDD		
N	*tm*	*mm*	*sm*	*tm*	*mm*	*tm*	*mm*	*sm*	*tm*	*mm*	*sm*
6	43.1	36 948	29 640	7	202 412	0.14	908	432	1.5	7 360	$4.83\ 10^{6}$
7	6 115	377 452	131 517	66	723 428	0.23	3 200	957	2.5	10 304	$6.28\ 10^{7}$
8	DNF	-	-	-	OOM	1	3 336	2 078	4	14 188	$8.16\ 10^{8}$
13	-	-	-	-	-	2 634	13 188	79 598	26	56 660	$3.02\ 10^{14}$
15	-	-	-	-	-	60 860	61 256		42	86 360	$5.11\ 10^{16}$
16	-	-	-	-	-	DNF	-	-	52	104 848	$6.65\ 10^{17}$
44	-	-	-	-	-	-	-	-	1143	$2.13\ 10^{6}$	$1.03\ 10^{49}$

Example: trains crossing...

Here, ITS embed Time Petri Nets

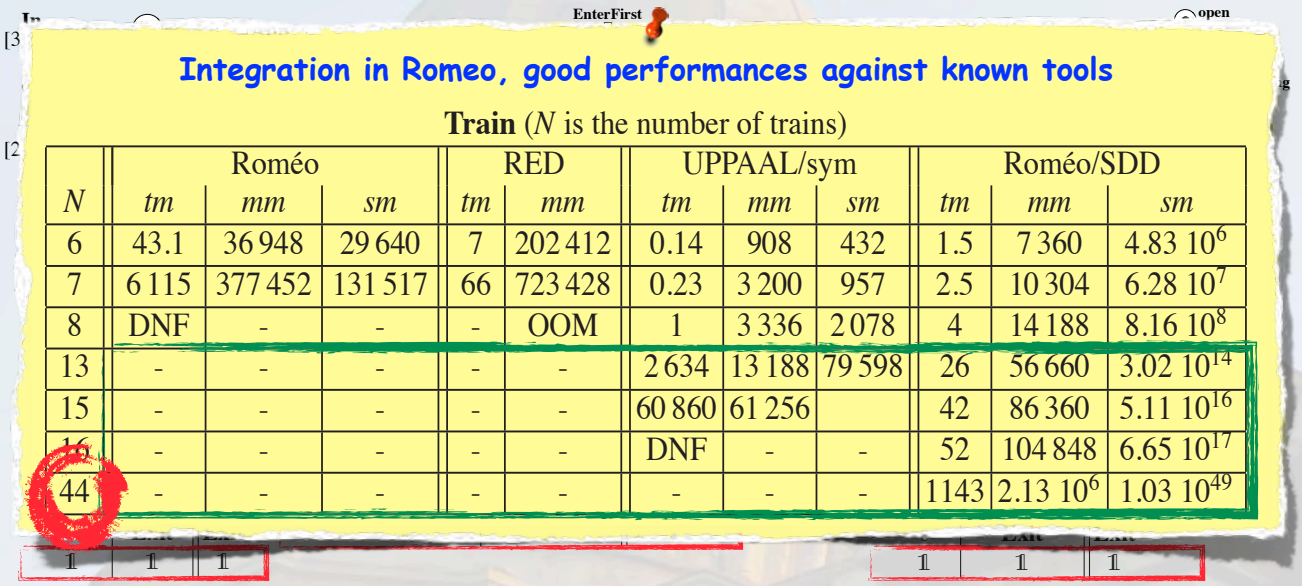
Integration in Romeo, good performances against known tools

Train (N is the number of trains)

	Roméo			RED		UPPAAL/sym			Roméo/SDD		
N	*tm*	*mm*	*sm*	*tm*	*mm*	*tm*	*mm*	*sm*	*tm*	*mm*	*sm*
6	43.1	36 948	29 640	7	202 412	0.14	908	432	1.5	7 360	$4.83\ 10^{6}$
7	6 115	377 452	131 517	66	723 428	0.23	3 200	957	2.5	10 304	$6.28\ 10^{7}$
8	DNF	-	-	-	OOM	1	3 336	2 078	4	14 188	$8.16\ 10^{8}$
13	-	-	-	-	-	2 634	13 188	79 598	26	56 660	$3.02\ 10^{14}$
15	-	-	-	-	-	60 860	61 256		42	86 360	$5.11\ 10^{16}$
16	-	-	-	-	-	DNF	-	-	52	104 848	$6.65\ 10^{17}$
44	-	-	-	-	-	-	-	-	1143	$2.13\ 10^{6}$	$1.03\ 10^{49}$

Example: trains crossing...

- Here, ITS embed Time Petri Nets



Integration in Romeo, good performances against known tools

Train (*N* is the number of trains)

	Roméo			RED		UPPAAL/sym			Roméo/SDD		
N	*tm*	*mm*	*sm*	*tm*	*mm*	*tm*	*mm*	*sm*	*tm*	*mm*	*sm*
6	43.1	36 948	29 640	7	202 412	0.14	908	432	1.5	7 360	$4.83\ 10^{6}$
7	6 115	377 452	131 517	66	723 428	0.23	3 200	957	2.5	10 304	$6.28\ 10^{7}$
8	DNF	-	-	-	OOM	1	3 336	2 078	4	14 188	$8.16\ 10^{8}$
13	-	-	-	-	-	2 634	13 188	79 598	26	56 660	$3.02\ 10^{14}$
15	-	-	-	-	-	60 860	61 256		42	86 360	$5.11\ 10^{16}$
[illegible]	-	-	-	-	-	DNF	-	-	52	104 848	$6.65\ 10^{17}$
44	-	-	-	-	-	-	-	-	1143	$2.13\ 10^{6}$	$1.03\ 10^{49}$

Towards Generation from a High-Level Language

- VeriSensor (Domain Specific Modeling Language)
  - Dedicated to sensor networks
  - Proposes relevant concepts
- Compilation into ITS
  - Seeking for the activation of the presented mechanisms
- Good scalability

Towards Generation from a High-Level Language

- VeriSensor (Domain Specific Modeling Language)
  - Dedicated to sensor networks
  - Proposes relevant concepts
- Compilation into ITS
  - Seeking for the activation of the presented mechanisms
- Good scalability

- More Tomorrow
  - Ph.D. Defense of Yann Ben Maissa
  - Tomorrow, 10h00, same place

Conclusion...

... and advertizing

Conclusions & Perspectives

- Combination of techniques to handle discrete time
  - Hierarchy + Symmetries + Decision Diagrams
    Shows good scalability so far!

- Relation with the System architecture
  - Structure can be exploited
  - Potential «assembly language» for Verification
    - Distributed systems
    - Timing constraints
- Implementation available
  - http://ddd.lip6.fr
    - Library & tool (graphic interface with Eclipse)
    - Reads Romeo & TINA nets...
  - Some of the presented tools embedded in the CosyVerif project
    - http://cosyverif.org

References

- [1loul 2003] F. Aloul, I. Markov, K. Sakallah. FORCE: a fast and easy-to-implement variable- ordering heuristic. In: ACM Great Lakes Symposium on VLSI, pp. 116–119. ACM, 2003
- [Ben Maïssa 2013] Y. Ben Maïssa, F. Kordon, S. Mouline and Y. Thierry-Mieg. Modeling and Analyzing Wireless Sensor Networks with VeriSensor: an Integrated Workflow. Transactions on Petri Nets and Other Models of Concurrency (ToPNoC), VIII, pages 24-47, Springer Verlag, 2013
- [Berard 2008] B. Bérard, S. Haddad, L. Hillah, F. Kordon, and Y. Thierry-Mieg. Collision avoidance in intelligent transport systems : towards an application of control theory. 9th International Workshop on Discrete Event Systems (WODES'08), pp 346-351, IEEE, 2008
- [Bryant 1986] R. Bryant. Graph-based algorithms for boolean function manipulation. IEEE Transactions on Computers 8:677–691, 1986
- [Clarke 1996] E. Clarke, R. Enders, T. Filkorn, and S. Jha, Exploiting symmetry in temporal logic model checking, Formal Methods in System Design, vol. 9, no. 1, pp. 77–104, 1996
- [Couvreur 2005] J.-M. Couvreur, Y. Thierry-Mieg. Hierarchical Decision Diagrams to Exploit Model Structure, 25th IFIP WG 6.1 International Conference on Formal Techniques for Networked and Distributed Systems (FORTE), LNCS, Taiwan, pp. 443-457, 2005
- [Heiner 2009] M. Heiner, M. Schwarick, A. Tovchigrechko. DSSZ-MC - A Tool for Symbolic Analysis of Extended Petri Nets. In PETRI NETS 2009. LNCS, vol. 5606, pp. 323–332. Springer
- [Hong 2012] S. Hong, F. Kordon, E. Paviot-Adet and S. Evangelista. Computing a Hierarchical Static order for Decision Diagram-Based Representation from P/T Nets. Transactions on Petri Nets and Other Models of Concurrency (ToPNoC), to appear, Springer Verlag, 2012
- [Thierry-Mieg 2009] Y. Thierry-Mieg, D. Poitrenaud, A. Hamez, and F. Kordon. Hierarchical Set Decision Diagrams and Regular Models. 15th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS), LNCS vol 5505, pages 1-15, Springer Verlag, March 2009
- [Thierry-Mieg 2011] Y. Thierry-Mieg, B. Bérard, F. Kordon, D. Lime, and O. H. Roux. Compositional Analysis of Discrete Time Petri nets. In 1st workshop on Petri Nets Compositions (CompoNet), vol 726, pages 17-31, CEUR, June 2011