States and Events in KandISTI
A Retrospective
in honour of Bernhard Steffen

Maurice H. ter Beek
ISTI–CNR, Pisa, Italy

joint work with

Alessandro Fantechi
UNIFI

Stefania Gnesi
ISTI–CNR

Franco Mazzanti
ISTI–CNR

B-Day

Limassol, Cyprus
4 November 2018
A long time ago in a galaxy far, far away...

According to Stefania and Alessandro

- The story started many years ago, more or less in the years following the success of CTL model checking [CES83/86]
- The process algebra world started thinking about developing action-based logics and verification frameworks
  - $\mu$-calculus
  - ACTL/ACTL*: Action-based CTL/CTL*
  - $\mu$-ACTL: ACTL with a fixpoint operator
  - AMC: ACTL Model Checker (with Franco)
  - FMC: Full $\mu$-calculus Model Checker (with Franco)
  - JACK: Just Another Concurrency (with Franco)
  - SAM: A Symbolic Model Checker for ACTL* (with Franco)
  - etc.

- The results of these efforts are at the basis of KandISTI

ISoLA’18: Franco’s 2nd consecutive RERS participation using KandISTI
A long time ago in a galaxy far, far away...

According to Stefania and Alessandro

- The story started many years ago, more or less in the years following the success of CTL model checking [CES83/86]
- The process algebra world started thinking about developing action-based logics and verification frameworks
  - $\mu$-calculus
  - ACTL/ACTL*: Action-based CTL/CTL*
  - $\mu$-ACTL: ACTL with a fixpoint operator
  - AMC: ACTL Model Checker (with Franco)
  - FMC: Full $\mu$-calculus Model Checker (with Franco)
  - JACK: Just Another Concurrency (with Franco)
  - SAM: A Symbolic Model Checker for ACTL* (with Franco)
  - etc.

- The results of these efforts are at the basis of KandISTI

ISoLA’18: Franco’s 2nd consecutive RERS participation using KandISTI
A long time ago in a galaxy far, far away...

According to Stefania and Alessandro

- The story started many years ago, more or less in the years following the success of CTL model checking [CES83/86]
- The process algebra world started thinking about developing action-based logics and verification frameworks
  - $\mu$-calculus
  - ACTL/ACTL*: Action-based CTL/CTL*
  - $\mu$-ACTL: ACTL with a fixpoint operator
  - AMC: ACTL Model Checker (with Franco)
  - FMC: Full $\mu$-calculus Model Checker (with Franco)
  - JACK: Just Another Concurrency (with Franco)
  - SAM: A Symbolic Model Checker for ACTL* (with Franco)
  - etc.

- The results of these efforts are at the basis of KandISTI

ISoLA’18: Franco’s 2nd consecutive RERS participation using KandISTI
According to Stefania and Alessandro

- The story started many years ago, more or less in the years following the success of CTL model checking [CES83/86]
- The process algebra world started thinking about developing action-based logics and verification frameworks
  
  - $\mu$-calculus
  - ACTL/ACTL*: Action-based CTL/CTL*
  - $\mu$-ACTL: ACTL with a fixpoint operator
  - AMC: ACTL Model Checker (with Franco)
  - FMC: Full $\mu$-calculus Model Checker (with Franco)
  - JACK: Just Another Concurrency (with Franco)
  - SAM: A Symbolic Model Checker for ACTL* (with Franco)
  - etc.

- The results of these efforts are at the basis of KandISTI

ISoLA’18: Franco’s 2nd consecutive RERS participation using KandISTI
This may be considered the first model-checking framework for process algebras.
This may be considered the first model-checking framework for process algebras.
In that period, Stefania, Alessandro and later Franco started working with the action-based logic ACTL, introduced by De Nicola & Vaandrager at LICS’90.
In that period, Stefania, Alessandro and later Franco started working with the action-based logic ACTL, introduced by De Nicola & Vaandrager at LICS’90.

**Abstract**

A system is described which supports proofs of both behavioural and logical properties of concurrent systems; these are specified by means of a process algebra and its associated logics. The logic is an action based version of the branching time logic CTL which we call ACTL; it is interpreted over transition labelled structures while CTL is interpreted over state labelled ones.
The next paper in the CAV’91 proceedings hints that this was perhaps the first time Stefania, Alessandro and Bernhard became aware of each other’s work.
Aarhus, 1989–90?

The next paper in the CAV’91 proceedings hints that this was perhaps the first time Stefania, Alessandro and Bernhard became aware of each other’s work...?
Leiden, 1990–96: Maurice studied Computer Science

2007 I first became aware of Bernhard when Tiziana pointed me to 'his' Kripke Transition Systems (KTS) during FMICS'07/SCP'11 work using Doubly-Labelled Transition Systems ($L^2$TS).

2011 Followed by my involvement in ISoLA: 3rd consecutive participation as track organiser (great networking opportunities! work with Rolf, Axel, etc.)

2012 Edited (with Stefania) 2 special issues in STTT on FMICS/AVoCS’16 and Formal Methods in Transport Systems (agreed with Bernhard at ISoLA’16).

2016 Edited (with Axel) a section in FoMaC on a hot topic.

2018 QSPL (IEEE TSE ’18 / FM ’18): this is about to appear, right Bernhard?

2018 Given Bernhard’s fundamental roles in ISoLA, STTT and FoMaC, I sure hope our roads will cross many more times...
2007 I first became aware of Bernhard when Tiziana pointed me to ‘his’ Kripke Transition Systems (KTS) during FMICS’07/SCP’11 work using Doubly-Labelled Transition Systems ($L^2$TS)

2011

2012 Followed by my involvement in ISoLA: 3rd consecutive participation as track organiser (great networking opportunities! work with Rolf, Axel, etc.)

2016 Edited (with Stefania) 2 special issues in STTT on FMICS/AVoCS’16 and 2018 Formal Methods in Transport Systems (agreed with Bernhard at ISoLA’16)

2016 Edited (with Axel, agreed at ISoLA’16) a section in FoMaC on a hot topic:

2018 QSPL (IEEE TSE’18 / FM’18): this is about to appear, right Bernhard?

2018 Given Bernhard’s fundamental roles in ISoLA, STTT and FoMaC, I sure 20?? hope our roads will cross many more times...
I first became aware of Bernhard when Tiziana pointed me to ‘his’ Kripke Transition Systems (KTS) during FMICS’07/SCP’11 work using Doubly-Labelled Transition Systems (L²TS): I often cite his SAS’99 paper since 😊

Followed by my involvement in ISoLA: 3rd consecutive participation as track organiser (great networking opportunities! work with Rolf, Axel, etc.)

Edited (with Stefania) 2 special issues in STTT on FMICS/AVoCS’16 and Formal Methods in Transport Systems (agreed with Bernhard at ISoLA’16)

Edited (with Axel, agreed at ISoLA’16) a section in FoMaC on a hot topic:

QSPL (IEEE TSE’18 / FM’18): this is about to appear, right Bernhard?

Given Bernhard’s fundamental roles in ISoLA, STTT and FoMaC, I sure hope our roads will cross many more times...
Leiden, 1990–96: Maurice studied Computer Science

2007 I first became aware of Bernhard when Tiziana pointed me to ‘his’ Kripke Transition Systems (KTS) during FMICS’07/SCP’11 work using Doubly-Labelled Transition Systems (L²TS): I often cite his SAS’99 paper since 😊

2011 Followed by my involvement in ISoLA: 3rd consecutive participation as track organiser (great networking opportunities! work with Rolf, Axel, etc.)

2012

2016 Edited (with Stefania) 2 special issues in STTT on FMICS/AVoCS’16 and Formal Methods in Transport Systems (agreed with Bernhard at ISoLA’16)

2018 Given Bernhard’s fundamental roles in ISoLA, STTT and FoMaC, I sure hope our roads will cross many more times...

2018 Edited (with Axel, agreed at ISoLA’16) a section in FoMaC on a hot topic: QSPL (IEEE TSE’18 / FM’18): this is about to appear, right Bernhard?


M.H. ter Beek et al. (CNR–ISTI / UNIFI) States and Events in KandISTI
Leiden, 1990–96: Maurice studied Computer Science

2007 I first became aware of Bernhard when Tiziana pointed me to ‘his’ Kripke Transition Systems (KTS) during FMICS’07/SCP’11 work using Doubly-Labelled Transition Systems (L²TS): I often cite his SAS’99 paper since 😊

2011

2012 Followed by my involvement in ISoLA: 3rd consecutive participation as track organiser (great networking opportunities! work with Rolf, Axel, etc.)

2016 Edited (with Stefania) 2 special issues in STTT on FMICS/AVoCS’16 and Formal Methods in Transport Systems (agreed with Bernhard at ISoLA’16)

2016 Edited (with Axel, agreed at ISoLA’16) a section in FoMaC on a hot topic:

2018 QSPL (IEEE TSE’18 / FM’18): this is about to appear, right Bernhard?

2018 Given Bernhard’s fundamental roles in ISoLA, STTT and FoMaC, I sure hope our roads will cross many more times...

2007 I first became aware of Bernhard when Tiziana pointed me to ‘his’ Kripke Transition Systems (KTS) during FMICS’07/SCP’11 work using Doubly-Labelled Transition Systems (L²TS): I often cite his SAS’99 paper since 😊

2011 Followed by my involvement in ISoLA: 3rd consecutive participation as track organiser (great networking opportunities! work with Rolf, Axel, etc.)

2016 Edited (with Stefania) 2 special issues in STTT on FMICS/AVoCS’16 and Formal Methods in Transport Systems (agreed with Bernhard at ISoLA’16)

2016 Edited (with Axel, agreed at ISoLA’16) a section in FoMaC on a hot topic:

2018 QSPL (IEEE TSE’18 / FM’18): this is about to appear, right Bernhard?

2018 Given Bernhard’s fundamental roles in ISoLA, STTT and FoMaC, I sure hope our roads will cross many more times...

Leiden, 1990–96: Maurice studied Computer Science

2007 I first became aware of Bernhard when Tiziana pointed me to ‘his’ Kripke Transition Systems (KTS) during FMICS’07/SCP’11 work using Doubly-Labelled Transition Systems (L²TS): I often cite his SAS’99 paper since 😊

2011

2012 Followed by my involvement in ISoLA: 3rd consecutive participation as track organiser (great networking opportunities! work with Rolf, Axel, etc.)

2016 Edited (with Stefania) 2 special issues in STTT on FMICS/AVoCS’16 and Formal Methods in Transport Systems (agreed with Bernhard at ISoLA’16)

2016 Edited (with Axel, agreed at ISoLA’16) a section in FoMaC on a hot topic:

2018 QSPL (IEEE TSE’18 / FM’18): this is about to appear, right Bernhard?

2018 Given Bernhard’s fundamental roles in ISoLA, STTT and FoMaC, I sure hope our roads will cross many more times...

____________________________________________________

Our Festschrift contribution

Brief overview on models and logics dealing with states and events

- KTS, L²TS, but also state/event systems of Graf & Loiseaux (TAPSOFT’93)
- μ-calculus, ACTL, but also SE-LTL and ARCTL

...followed by specificities of our KandISTI model checker and logic

http://fmt.isti.cnr.it/kandisti

Family of model checkers developed at ISTI–CNR for over 2 decades
- FMC (PDPTA’99), UMC (SCP’11), CMC (ACM TOSEM’12), VMC (FM’12)

Explicit-state on-the-fly model checking of properties in state-based and event-based branching-time temporal logics, building on (A)CTL
- e.g. UCTL (FMICS’07), SocL (FASE’08), v-ACTL (JLAMP’16)

Complexity is linear w.r.t. size of the model and size of the formula

R. Cleaveland, B. Steffen, A Linear-Time Model-Checking Algorithm for the Alternation-Free Modal Mu-Calculus. FMSD’93
Our Festschrift contribution

Brief overview on models and logics dealing with states and events

- KTS, $L^2TS$, but also state/event systems of Graf & Loiseaux (TAPSOFT'93)
- $\mu$-calculus, ACTL, but also SE-LTL and ARCTL

...followed by specificities of our KandISTI model checker and logic

http://fmt.isti.cnr.it/kandisti

Family of model checkers developed at ISTI–CNR for over 2 decades
- FMC (PDPTA’99), UMC (SCP’11), CMC (ACM TOSEM’12), VMC (FM’12)

Explicit-state on-the-fly model checking of properties in state-based and event-based branching-time temporal logics, building on (A)CTL
- e.g. UCTL (FMICS’07), SocL (FASE’08), $v$-ACTL (JLAMP’16)

Complexity is linear w.r.t. size of the model and size of the formula

R. Cleaveland, B. Steffen, A Linear-Time Model-Checking Algorithm for the Alternation-Free Modal Mu-Calculus. FMSD’93
Brief overview on models and logics dealing with states and events

- KTS, $L^2TS$, but also state/event systems of Graf & Loiseaux (TAPSOFT’93)
- $\mu$-calculus, ACTL, but also SE-LTL and ARCTL

...followed by specificities of our KandISTI model checker and logic

http://fmt.isti.cnr.it/kandisti

Family of model checkers developed at ISTI–CNR for over 2 decades

FMC (PDPTA’99), UMC (SCP’11), CMC (ACM TOSEM’12), VMC (FM’12)

Explicit-state on-the-fly model checking of properties in state-based and event-based branching-time temporal logics, building on (A)CTL

e.g. UCTL (FMICS’07), SocL (FASE’08), v-ACTL (JLAMP’16)

Complexity is linear w.r.t. size of the model and size of the formula

R. Cleaveland, B. Steffen, A Linear-Time Model-Checking Algorithm for the Alternation-Free Modal Mu-Calculus. FMSD’93
Our Festschrift contribution

Brief overview on models and logics dealing with states and events

- KTS, L²TS, but also state/event systems of Graf & Loiseaux (TAPSOFT’93)
- μ-calculus, ACTL, but also SE-LTL and ARCTL

...followed by specificities of our KandISTI model checker and logic

http://fmt.isti.cnr.it/kandisti

Family of model checkers developed at ISTI–CNR for over 2 decades

FMC (PDPTA’99), UMC (SCP’11), CMC (ACM TOSEM’12), VMC (FM’12)

Explicit-state on-the-fly model checking of properties in state-based and event-based branching-time temporal logics, building on (A)CTL
e.g. UCTL (FMICS’07), SocL (FASE’08), v-ACTL (JLAMP’16)

Complexity is linear w.r.t. size of the model and size of the formula

R. Cleaveland, B. Steffen, A Linear-Time Model-Checking Algorithm for the Alternation-Free Modal Mu-Calculus. FMSD’93

M.H. ter Beek et al. (CNR–ISTI / UNIFI) States and Events in KandISTI B-Day 5 / 12
Our Festschrift contribution

Brief overview on models and logics dealing with states and events

- KTS, L²TS, but also state/event systems of Graf & Loiseaux (TAPSOFT’93)
- μ-calculus, ACTL, but also SE-LTL and ARCTL

...followed by specificities of our KandISTI model checker and logic

http://fmt.isti.cnr.it/kandisti

Family of model checkers developed at ISTI–CNR for over 2 decades

- FMC (PDPTA’99), UMC (SCP’11), CMC (ACM TOSEM’12), VMC (FM’12)

Explicit-state on-the-fly model checking of properties in state-based and event-based branching-time temporal logics, building on (A)CTL

  - e.g. UCTL (FMICS’07), SocL (FASE’08), v-ACTL (JLAMP’16)

Complexity is linear w.r.t. size of the model and size of the formula

R. Cleaveland, B. Steffen, A Linear-Time Model-Checking Algorithm for the Alternation-Free Modal Mu-Calculus. FMSD’93
KandISTI architecture

Logical verification engine shared by all tools observes the underlying model as an abstract L^2TS independent from the operational semantics of the tool’s specification language, thanks to an associated set of abstraction rules.

M.H. ter Beek, S. Gnesi, F. Mazzanti, From EU Projects to a Family of Model Checkers: From Kandinsky to KandISTI, Festschrift Martin Wirsing, 2015
Explicit abstraction mechanism allows to specify which details of the model become observable labels on the $L^2TS'$ states and transitions.

M.H. ter Beek, S. Gnesi, F. Mazzanti, From EU Projects to a Family of Model Checkers: From Kandinsky to KandISTI, Festschrift Martin Wirsing, 2015
KandISTI logic ($L^2TS$ semantics)

- Parametric state predicates (state labels)
  e.g. $\text{pred}_1(\text{arg}_1, \text{arg}_2)$, $\text{pred}_2$, $\text{pred}_3(\ast, \text{arg}_3)$

- Parametric event formulae (Bool expressions over transition labels)
  e.g. $(\text{act}_1(\text{arg}_1, \text{arg}_2) \text{ or } \text{act}_2)$, not $\text{act}_3(\text{arg}_3, \ast, \ast)$

- Box, Diamond, fixpoint operators (i.e. full modal $\mu$-calculus)
  e.g. $\text{max}\ Y: \text{max}\ Z : ( (\langle \text{act}_2(\text{arg}_1) \rangle\ Y) \text{ or } \langle \text{act}_2 \rangle\ Z)$

- CTL operators (e.g. neXt, Always, (Weak) Until, Globally, Eventually)
  e.g. $\text{EX}\ \text{pred}_1$, $\text{A}[\text{pred}_1(\text{arg}_1)\ U\ \text{pred}_2]$, $\text{AG}\ \text{EF}\ \text{pred}_1$

- ACTL-like operators (i.e. event-based variants of CTL operators)
  e.g. $\text{EX}\ \{\text{act}_1\}\ \text{true}$, $\text{A}\ [\text{pred}_1(\text{arg}_1)\ \{\text{act}_1\}\ U\ \{\text{act}_2\}\ \text{pred}_2]$

- Parametric formulae that express data correlations
  e.g. $[\text{act}_1(\$1, \$2)]\ \text{AF}\ \{\text{act}_2(\%1, \%2)\}\ \text{true}$, $\text{EF}\ \{\$1\}\ \text{EF}\ \{\%1\}\ \text{true}$

- Deontic variants of some of the above operators (for MTS, next slide)
  e.g. $\langle\text{act}_1\rangle\#\ \text{true}$, $\text{EF}\#\ \{\text{act}\}\ \text{pred}_1$

- Special-purpose predefined state predicates
  e.g. $\text{PRINT}(\text{msg}, \text{arg}_1, \text{arg}_2)$, $\text{DEPTH}_\text{LT}_n$, $\text{FINAL}$
KandISTI logic ($L^2$TS semantics)

- Parametric state predicates (state labels)
  
  \[ \text{e.g. } \text{pred1}(\text{arg1}, \text{arg2}), \text{pred2}, \text{pred3}(\ast, \text{arg3}) \]

- Parametric event formulae (Bool expressions over transition labels)
  
  \[ \text{e.g. } (\text{act1}(\text{arg1}, \text{arg2}) \text{ or } \text{act2}), \ \text{not } \text{act3}(\text{arg3}, \ast, \ast) \]

- Box, Diamond, fixpoint operators (i.e. full modal $\mu$-calculus)
  
  \[ \text{e.g. } \text{max } Y : \text{max } Z : ( (\langle \text{act2}(\text{arg1}) \rangle Y ) \text{ or } \langle \text{act2} \rangle Z ) \]

- CTL operators (e.g. neXt, Always, (Weak) Until, Globally, Eventually)
  
  \[ \text{e.g. } EX \text{ pred1}, \ A[\text{pred1}(\text{arg1}) \ \text{U pred2}], \ AG \ EF \ \text{pred1} \]

- ACTL-like operators (i.e. event-based variants of CTL operators)
  
  \[ \text{e.g. } EX \{\text{act1}\} \text{ true}, \ A[\text{pred1}(\text{arg1}) \ \{\text{act1}\} \text{ U}\{\text{act2}\} \ \text{pred2}] \]

- Parametric formulae that express data correlations
  
  \[ \text{e.g. } [\text{act1}(\$1, \$2)] \text{ AF}\{\text{act2}(\%1, \%2)\} \text{ true}, \ EF\{\$1\} \ EF\{\%1\} \text{ true} \]

- Deontic variants of some of the above operators (for MTS, next slide)
  
  \[ \text{e.g. } \langle \text{act1} \rangle \# \text{ true}, \ EF\#\{\text{act}\} \ \text{pred1} \]

- Special-purpose predefined state predicates
  
  \[ \text{e.g. } \text{PRINT}(\text{msg}, \text{arg1}, \text{arg2}), \ \text{DEPTH_LT}_n, \ \text{FINAL} \]
KandISTI logic ($L^2TS$ semantics)

- **Parametric state predicates (state labels)**
  
  e.g. $\text{pred}_1(\text{arg}_1, \text{arg}_2)$, $\text{pred}_2$, $\text{pred}_3(\ast, \text{arg}_3)$

- **Parametric event formulae (Bool expressions over transition labels)**
  
  e.g. $(\text{act}_1(\text{arg}_1, \text{arg}_2) \text{ or } \text{act}_2)$, $\text{not } \text{act}_3(\text{arg}_3, \ast, \ast)$

- **Box, Diamond, fixpoint operators (i.e. full modal $\mu$-calculus)**
  
  e.g. $\max Y: \max Z: (((\langle \text{act}_2(\text{arg}_1) \rangle Y) \text{ or } \langle \text{act}_2 \rangle Z)$

- **CTL operators (e.g. neXt, Always, (Weak) Until, Globally, Eventually)**
  
  e.g. $\text{EX } \text{pred}_1$, $A[\text{pred}_1(\text{arg}_1) U \text{pred}_2]$, $AG EF \text{pred}_1$

- **ACTL-like operators (i.e. event-based variants of CTL operators)**
  
  e.g. $\text{EX } \{\text{act}_1\} \text{true}$, $A[\text{pred}_1(\text{arg}_1) \{\text{act}_1\} U \{\text{act}_2\} \text{pred}_2]$

- **Parametric formulae that express data correlations**
  
  e.g. $[\text{act}_1(\$1, \$2)] A\text{F } \{\text{act}_2(\%1, \%2)\} \text{true}$, $EF \{\$1\} EF \{\%1\} \text{true}$

- **Deontic variants of some of the above operators (for MTS, next slide)**
  
  e.g. $\langle \text{act}_1 \rangle \# \text{true}$, $EF \# \{\text{act}\} \text{pred}_1$

- **Special-purpose predefined state predicates**
  
  e.g. $\text{PRINT}(\text{msg}, \text{arg}_1, \text{arg}_2)$, $\text{DEPTH_LT}_n$, $\text{FINAL}$
KandISTI logic ($L^2TS$ semantics)

- Parametric state predicates (state labels)
  e.g. $\text{pred1}(\text{arg1}, \text{arg2}), \text{pred2}, \text{pred3}(*, \text{arg3})$

- Parametric event formulae (Bool expressions over transition labels)
  e.g. $(\text{act1}(\text{arg1}, \text{arg2}) \text{ or } \text{act2}), \text{ not } \text{act3}(\text{arg3}, *, *)$

- Box, Diamond, fixpoint operators (i.e. full modal $\mu$-calculus)
  e.g. $\max Y: \max Z: ((\langle \text{act2}(\text{arg1}) \rangle Y) \text{ or } \langle \text{act2} \rangle Z)$

- CTL operators (e.g. neXt, Always, (Weak) Until, Globally, Eventually)
  e.g. $EX \text{ pred1}, A[\text{pred1}(\text{arg1}) U \text{ pred2}], AG EF \text{ pred1}$

- ACTL-like operators (i.e. event-based variants of CTL operators)
  e.g. $EX \{\text{act1}\} true, A[\text{pred1}(\text{arg1}) \{\text{act1}\} U \{\text{act2}\} \text{ pred2}]$

- Parametric formulae that express data correlations
  e.g. $[\text{act1}($$1, $2$)] AF \{\text{act2}(%1, %2)\} true, EF \{$$1$\} EF \{%1\} true$

- Deontic variants of some of the above operators (for MTS, next slide)
  e.g. $\langle \text{act1}\rangle# true, EF# \{\text{act}\} \text{ pred1}$

- Special-purpose predefined state predicates
  e.g. $\text{PRINT}(\text{msg}, \text{arg1}, \text{arg2}), \text{ DEPTH_LT}_n, \text{ FINAL}$
KandISTI logic ($L^2TS$ semantics)

- Parametric state predicates (state labels)
  
  e.g. \(\text{pred1}(\text{arg1}, \text{arg2}), \text{pred2}, \text{pred3}(\ast, \text{arg3})\)

- Parametric event formulae (Bool expressions over transition labels)
  
  e.g. \((\text{act1}(\text{arg1}, \text{arg2}) \text{ or } \text{act2}), \text{ not } \text{act3}(\text{arg3}, \ast, \ast)\)

- Box, Diamond, fixpoint operators (i.e. full modal $\mu$-calculus)
  
  e.g. \(\text{max } Y: \text{max } Z: ((\langle \text{act2}(\text{arg1}) \rangle Y) \text{ or } \langle \text{act2} \rangle Z)\)

- CTL operators (e.g. neXt, Always, (Weak) Until, Globally, Eventually)
  
  e.g. \(\text{EX } \text{pred1}, A[\text{pred1}(\text{arg1}) \ U \ \text{pred2}], \ AG \ EF \ \text{pred1}\)

- ACTL-like operators (i.e. event-based variants of CTL operators)
  
  e.g. \(\text{EX } \{\text{act1}\} \ true, A[\text{pred1}(\text{arg1}) \ \{\text{act1}\} \ U \{\text{act2}\} \ \text{pred2}]\)

- Parametric formulae that express data correlations
  
  e.g. \([\text{act1}($1, $2)] \ A F \{\text{act2}(%1,%2)\} \ true, EF \{1\} \ EF \{1\} \ true\)

- Deontic variants of some of the above operators (for MTS, next slide)
  
  e.g. \(\langle \text{act1}\rangle \# \ true, EF\# \{\text{act}\} \ \text{pred1}\)

- Special-purpose predefined state predicates
  
  e.g. \(\text{PRINT}(\text{msg}, \text{arg1}, \text{arg2}), \ DEPTH_LT_n, \ FINAL\)
KandISTI logic ($L^2TS$ semantics)

- Parametric state predicates (state labels)
  e.g. $pred1(arg1,arg2)$, $pred2$, $pred3(*,arg3)$

- Parametric event formulae (Bool expressions over transition labels)
  e.g. $(act1(arg1,arg2) \lor act2)$, $not act3(arg3,*,*)$

- Box, Diamond, fixpoint operators (i.e. full modal $\mu$-calculus)
  e.g. $max Y: max Z: (\langle act2(arg1) \rangle Y) or \langle act2 \rangle Z$

- CTL operators (e.g. neXt, Always, (Weak) Until, Globally, Eventually)
  e.g. $EX pred1$, $A[pred1(arg1) U pred2]$, $AG EF pred1$

- ACTL-like operators (i.e. event-based variants of CTL operators)
  e.g. $EX \{act1\} true$, $A[pred1(arg1) \{act1\} U \{act2\} pred2]$

- Parametric formulae that express data correlations
  e.g. $[act1($1, $2)] AF \{act2($1,$2)\} true$, $EF \{$1$\} EF \{$1\} true$

- Deontic variants of some of the above operators (for MTS, next slide)
  e.g. $\langle act1 \rangle \# true$, $EF \# \{act\} pred1$

- Special-purpose predefined state predicates
  e.g. $PRINT(msg,arg1,arg2)$, $DEPTH_LT_n$, $FINAL$
KandISTI logic (L²TS semantics)

- Parametric state predicates (state labels)
  e.g. pred1(arg1, arg2), pred2, pred3(*, arg3)

- Parametric event formulae (Bool expressions over transition labels)
  e.g. (act1(arg1, arg2) or act2), not act3(arg3, *, *)

- Box, Diamond, fixpoint operators (i.e. full modal $\mu$-calculus)
  e.g. max Y: max Z: ((⟨act2(arg1)⟩ Y) or ⟨act2⟩ Z)

- CTL operators (e.g. neXt, Always, (Weak) Until, Globally, Eventually)
  e.g. EX pred1, A[pred1(arg1) U pred2], AG EF pred1

- ACTL-like operators (i.e. event-based variants of CTL operators)
  e.g. EX {act1} true, A[pred1(arg1) {act1} U {act2} pred2]

- Parametric formulae that express data correlations
  e.g. [act1($1, 2$)] AF {act2(%1, %2)} true, EF {$1} EF {%1} true

- Deontic variants of some of the above operators (for MTS, next slide)
  e.g. ⟨act1⟩# true, EF# {act} pred1

- Special-purpose predefined state predicates
  e.g. PRINT(msg, arg1, arg2), DEPTH_LT_n, FINAL
KandISTI logic ($L^2$TS semantics)

- Parametric state predicates (state labels)
  e.g. $\text{pred1}(\text{arg1}, \text{arg2}), \text{pred2}, \text{pred3}(\ast, \text{arg3})$

- Parametric event formulae (Bool expressions over transition labels)
  e.g. $(\text{act1}(\text{arg1}, \text{arg2}) \text{ or } \text{act2}), \text{not} \ \text{act3}(\text{arg3}, \ast, \ast)$

- Box, Diamond, fixpoint operators (i.e. full modal $\mu$-calculus)
  e.g. $\text{max } Y: \text{max } Z: (\langle \text{act2}(\text{arg1}) \rangle Y) \text{ or } \langle \text{act2} \rangle Z$

- CTL operators (e.g. neXt, Always, (Weak) Until, Globally, Eventually)
  e.g. $EX \ \text{pred1}, \ A[\text{pred1}(\text{arg1}) \ U \ \text{pred2}], \ AG \ EF \ \text{pred1}$

- ACTL-like operators (i.e. event-based variants of CTL operators)
  e.g. $EX \ \{\text{act1}\} \ true, \ A[\text{pred1}(\text{arg1}) \ \{\text{act1}\} \ U \ \{\text{act2}\} \ \text{pred2}]$

- Parametric formulae that express data correlations
  e.g. $[\text{act1}($1, $2)] \ AF \ \{\text{act2}(\%1,\%2)\} \ true, \ EF \ \{$1\} \ EF \ \{\%1\} \ true$

- Deontic variants of some of the above operators (for MTS, next slide)
  e.g. $\langle \text{act1}\rangle \# \ true, \ EF \# \ \{\text{act}\} \ \text{pred1}$

- Special-purpose predefined state predicates
  e.g. $PRINT(\text{msg}, \text{arg1}, \text{arg2}), \ DEPTH_LT_n, \ FINAL$
KandISTI logic ($L^2$TS semantics)

- Parametric state predicates (state labels)
  e.g. \( \text{pred1}(\text{arg1}, \text{arg2}), \text{pred2}, \text{pred3}(\ast, \text{arg3}) \)

- Parametric event formulae (Bool expressions over transition labels)
  e.g. \( (\text{act1}(\text{arg1}, \text{arg2}) \text{ or } \text{act2}), \text{not act3}(\text{arg3}, \ast, \ast) \)

- Box, Diamond, fixpoint operators (i.e. full modal $\mu$-calculus)
  e.g. \( \max Y: \max Z: (((\langle \text{act2}(\text{arg1}) \rangle Y) \text{ or } \langle \text{act2} \rangle Z) \)

- CTL operators (e.g. neXt, Always, (Weak) Until, Globally, Eventually)
  e.g. \( EX \text{ pred1}, A[pred1(\text{arg1}) U pred2], AG EF pred1 \)

- ACTL-like operators (i.e. event-based variants of CTL operators)
  e.g. \( EX \{\text{act1}\} true, A[pred1(\text{arg1}) \{\text{act1}\} U \{\text{act2}\} \text{ pred2}] \)

- Parametric formulae that express data correlations
  e.g. \( [\text{act1}(\$1, \$2)] AF \{\text{act2}(\%1, \%2)\} true, EF \{\$1\} EF \{\%1\} true \)

- Deontic variants of some of the above operators (for MTS, next slide)
  e.g. \( \langle \text{act1} \rangle \# true, EF \# \{\text{act}\} \text{ pred1} \)

- Special-purpose predefined state predicates
  e.g. \( \text{PRINT}(\text{msg}, \text{arg1}, \text{arg2}), \text{DEPTH}_{LT_n}, \text{FINAL} \)
Application: Software Product Lines

Modal Transition Systems (MTS) by Larsen & Thomsen at LICS’88
- LTS distinguishing possible (may) and required (must) transitions

MTS with variability constraints (MTS_υ) by the four of us in JLAMP’16
- additional variability constraints (mimicking feature models) to be able to decide which implementations (LTS) are product variants
Application: Software Product Lines

Modal Transition Systems (MTS) by Larsen & Thomsen at LICS’88
- LTS distinguishing possible (may) and required (must) transitions

MTS with variability constraints (MTS$_v$) by the four of us in _JLAMP_’16
- additional variability constraints (mimicking feature models) to be able to decide which implementations (LTS) are product variants
Modal Transition Systems (MTS) by Larsen & Thomsen at LICS’88
- LTS distinguishing possible (may) and required (must) transitions

MTS with variability constraints ($\text{MTS}_v$) by the four of us in JLAMP’16
- additional variability constraints (mimicking feature models) to be able to decide which implementations (LTS) are product variants

MTS + v-ACTL can be interpreted as $L^2TS + \text{ACTL}$:

$$T = a.T + b(\text{may}).\text{nil}$$

$$\langle a \rangle \not\equiv \text{true}$$

$$\langle a \land \neg \text{may} \rangle \equiv \text{true}$$

From a v-ACTL formula and an MTS representation (left) of a process (middle) to a corresponding ACTL formula over the $L^2TS$ interpretation (right) of the MTS
Application: Software Product Lines

Modal Transition Systems (MTS) by Larsen & Thomsen at LICS’88
- LTS distinguishing possible (may) and required (must) transitions

MTS with variability constraints (MTS_υ) by the four of us in JLAMP’16
- additional variability constraints (mimicking feature models) to be able to decide which implementations (LTS) are product variants

An MTS and four implementation variants (LTS):

```
\begin{align*}
\sim p & \xrightarrow{b} q \\
\sim a & \xrightarrow{r} s \\
\end{align*}
```

```
\begin{align*}
\sim p & \\
\sim a & \xrightarrow{r} s \\
\end{align*}
```

```
\begin{align*}
\sim p & \xrightarrow{b} q \\
\sim a & \xrightarrow{r} s \\
\end{align*}
```

```
\begin{align*}
\sim p & \xrightarrow{b} q \\
\sim a & \xrightarrow{r} s \\
\end{align*}
```

With variability constraint $a \text{ ALT } b$, only the two central LTS are product variants
Modal Transition Systems (MTS) by Larsen & Thomsen at LICS’88
- LTS distinguishing possible (may) and required (must) transitions

MTS with variability constraints (MTS$\nu$) by the four of us in JLAMP’16
- additional variability constraints (mimicking feature models) to be able to decide which implementations (LTS) are product variants

An MTS and four implementation variants (LTS):

With variability constraint $a$ ALT $b$, only the two central LTS are product variants

In SCP’19 we prove MTS$\nu$ equally expressive as Featured Transition Systems (FTS) introduced by Axel et al. at ICSE’10 and IEEE TSE’13
VMC: Variability Model Checker

Aim: lift known formal methods and tools from single system (product) to a set of products (family) by exploiting variability modelling and analysis

VMC offers both product-based and family-based variability analyses

1. The set of all product variants can explicitly be generated and the resulting LTS verified against a logic property (expressed in ACTL)

2. A logic property (expressed in v-ACTL) can be verified on the MTS, relying on the fact that under certain conditions its validity implies validity of the same property for all its product variants (next slide)
VMC: Variability Model Checker

Aim: lift known formal methods and tools from single system (product) to a set of products (family) by exploiting variability modelling and analysis

VMC offers both product-based and family-based variability analyses

1. The set of all product variants can explicitly be generated and the resulting LTS verified against a logic property (expressed in ACTL)

2. A logic property (expressed in v-ACTL) can be verified on the MTS, relying on the fact that under certain conditions its validity implies validity of the same property for all its product variants (next slide)
Aim: lift known formal methods and tools from single system (product) to a set of products (family) by exploiting variability modelling and analysis

VMC offers both product-based and family-based variability analyses

1. The set of all product variants can explicitly be generated and the resulting LTS verified against a logic property (expressed in ACTL)

2. A logic property (expressed in v-ACTL) can be verified on the MTS, relying on the fact that under certain conditions its validity implies validity of the same property for all its product variants (next slide)
Aim: lift known formal methods and tools from single system (product) to a set of products (family) by exploiting variability modelling and analysis. VMC offers both product-based and family-based variability analyses.

1. The set of all product variants can explicitly be generated and the resulting LTS verified against a logic property (expressed in ACTL).

2. A logic property (expressed in v-ACTL) can be verified on the MTS, relying on the fact that under certain conditions its validity implies validity of the same property for all its product variants (next slide).
Preservation of formulae in v-ACTL

v-ACTL\(\Box/v\)-ACTL\(\Box\):

\[
\phi ::= \text{false} \mid \text{true} \mid \phi \text{ and } \phi \mid \phi \text{ or } \phi \mid [\chi] \phi \mid \langle \chi \rangle \# \phi \mid EX \# \phi \mid EX \# \{\chi\} \phi \mid EF \# \phi \mid EF \# \{\chi\} \phi \mid AF \# \phi \mid AF \# \{\chi\} \phi \mid AG \phi \mid AF \phi \mid AF \{\chi\} \phi
\]

any formula that is true for MTS\(\nu\), is also true for all products (LTS)

v-ACTL\(\neg\):

\[
\psi ::= \text{false} \mid \text{true} \mid \psi \text{ and } \psi \mid \psi \text{ or } \psi \mid [\chi] \# \psi \mid \langle \chi \rangle \psi \mid EX \psi \mid EX \{\chi\} \psi \mid EF \psi \mid EF \{\chi\} \psi \mid AF \psi \mid AF \{\chi\} \psi
\]

any formula that is false for MTS, is also false for all products (LTS)
Preservation of formulae in v-ACTL

\(\text{v-ACTL} \Box / \text{v-ACTLive} \Box:\)

\[\phi ::= \text{false} \mid \text{true} \mid \phi \text{ and } \phi \mid \phi \text{ or } \phi \mid [\chi] \phi \mid \langle \chi \rangle \# \phi \mid EX \# \phi \mid EX \# \{\chi\} \phi \mid EF \# \phi \mid EF \# \{\chi\} \phi \mid AF \# \phi \mid AF \# \{\chi\} \phi \mid AG \phi \mid AF \phi \mid AF \{\chi\} \phi\]

any formula that is true for MTS\(\nu\), is also true for all products (LTS)

\(\text{v-ACTL}^-:\)

\[\psi ::= \text{false} \mid \text{true} \mid \psi \text{ and } \psi \mid \psi \text{ or } \psi \mid [\chi] \# \psi \mid \langle \chi \rangle \psi \mid EX \psi \mid EX \{\chi\} \psi \mid EF \psi \mid EF \{\chi\} \psi \mid AF \phi \mid AF \{\chi\} \phi\]

any formula that is false for MTS, is also false for all products (LTS)
Live states use SPL-specific information

\[ S \models \phi \Rightarrow S_p \models \phi \quad \forall \text{product LTS } S_p \text{ of } MTS_\nu \text{ } S \]

Recall: all (reachable) must transitions are preserved in product LTS.

Live action sets define live states (do not occur as final in any product).

In any product in which \( p \) occurs, \( p \) has at least one outgoing transition.

\[ p \text{ is a live state, since } a \text{ OR } b \text{ gives rise to a live action set } \{ a, b \} \]
Live states use SPL-specific information

\[ S \models \phi \Rightarrow S_p \models \phi \quad \forall \text{product LTS } S_p \text{ of } MTS \]

Recall: all (reachable) must transitions are preserved in product LTS

Live action sets define live states (do not occur as final in any product)

In any product in which \( p \) occurs, \( p \) has at least one outgoing transition

\[ \Rightarrow p \text{ is a live state, since } a \text{ OR } b \text{ gives rise to a live action set } \{ a, b \} \]
Live states use SPL-specific information

\[ S \models \phi \Rightarrow S_p \models \phi \quad \forall \text{product LTS } S_p \text{ of } MTS_\nu \quad S \]

Recall: all (reachable) must transitions are preserved in product LTS

Live action sets define live states (do not occur as final in any product)

\[
\begin{array}{c}
\text{MTS}_\nu \\
\text{Constraint}
\end{array}
\]

\[
\begin{array}{c}
a \text{ OR } b
\end{array}
\]

In any product in which \( p \) occurs, \( p \) has at least one outgoing transition

\[ \Rightarrow p \text{ is a live state, since } a \text{ OR } b \text{ gives rise to a live action set } \{ a, b \} \]
Live states use SPL-specific information

\[ S \models \phi \Rightarrow S_p \models \phi \quad \forall \text{ product LTS } S_p \text{ of } MTS \upsilon \ S \]

Recall: all (reachable) must transitions are preserved in product LTS

Live action sets define live states (do not occur as final in any product)

In any product in which \( p \) occurs, \( p \) has at least one outgoing transition

\[ \Rightarrow p \text{ is a live state, since } a \text{ OR } b \text{ gives rise to a live action set } \{ a, b \} \]
Live states use SPL-specific information

\[ S \models \phi \Rightarrow S_p \models \phi \quad \forall \text{product LTS } S_p \text{ of MTS}_S \]

Recall: all (reachable) must transitions are preserved in product LTS

Live action sets define live states (do not occur as final in any product)

In any product in which \( p \) occurs, \( p \) has at least one outgoing transition

\( \Rightarrow p \) is a live state, since \( a \) OR \( b \) gives rise to a live action set \( \{a, b\} \)
Thanks!

From Pisa we wish Bernhard many more...

- papers
- STTT’s
- ISoLA’s
- FoMaC’s
- etc.
- but above all ...
- many more birthdays!!!
- in good health
- with family and friends!
From Pisa we wish Bernhard many more...

- papers
- STTT’s
- ISoLA’s
- FoMaC’s
- etc.

- but above all ...
- many more birthdays!!!
- in good health
- with family and friends!
Thanks!

From Pisa we wish Bernhard many more...

- papers
- STTT’s
- ISoLA’s
- FoMaC’s
- etc.
- but above all ...
  - many more birthdays!!!
  - in good health
  - with family and friends!
Thanks!

From Pisa we wish Bernhard many more...

- papers
- STTT’s
- ISoLA’s
- FoMaC’s
- etc.
- but above all ...
- many more birthdays!!!
  - in good health
  - with family and friends!
From Pisa we wish Bernhard many more...

- papers
- STTT’s
- ISoLA’s
- FoMaC’s
- etc.
- but above all ...
- many more birthdays!!!
- in good health
- with family and friends!
Thanks!

From Pisa we wish Bernhard many more...

- papers
- STTT’s
- ISoLA’s
- FoMaC’s
- etc.
- but above all ...
- many more birthdays!!!
- in good health
- with family and friends!