A Compositional Algebra of Specifications

Uli Fahrenberg

IRISA/Inria Rennes

Joint work with N. Beneš, B. Delahaye, J. Křetinský, A. Legay, L.-M. Traonouez

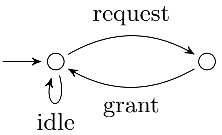
QuantLA 2014

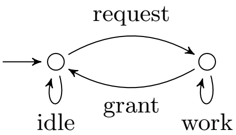
Specifications

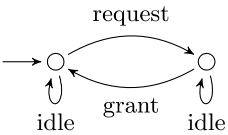
- Specification = property (of a formal model of a system)
- Example:

$$\mathsf{AG}\big(\text{request} \Rightarrow \mathsf{AX}(\text{work } \mathsf{AW} \text{ grant})\big)$$

  "after a request, only work is allowed, until grant is executed"

- satisfied by 

  and 

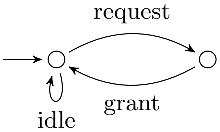
- not satisfied by 

Operations on specifications

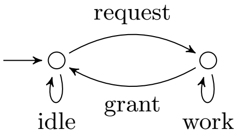
- logical operations: conjunction, disjunction, negation
- implication / refinement / strengthening

$$\mathsf{AG}\big(\text{request} \Rightarrow \text{grant}\big) \le \mathsf{AG}\big(\text{request} \Rightarrow \mathsf{AX}(\text{work}\ \mathsf{AW}\ \text{grant})\big)$$

- satisfied by



- not satisfied by



Model checking

- Algorithm for deciding whether or not a model satisfies a specification
- Popular specification formalisms: CTL, LTL, CTL*, μ-calculus
- Successful tools: Cadence SMV, Java Pathfinder, NuSMV, Spin, ...
- But: state space explosion 
- No chance to model-check industrial-size systems!
- Different approach needed
- For example: compositionality

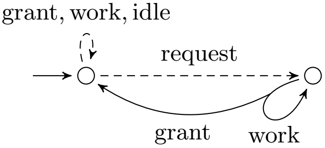
Compositionality

- Idea: Model check large systems by checking one component at a time
  - if $C_1 \models S_1$ and $C_2 \models S_2$ and ...
  - then $C_1 \| C_2 \| \dots \models S_1 \| S_2 \| \dots$
- Needs operation of structural composition $\|$ on models and specifications
- Also useful: decomposition
  - if $C_1 \models S_1$ and $C_1 \| C_2 \models S$
  - synthesize property S_2 so that $C_2 \models S_2$

Disjunctive modal transition systems

CTL $\mathsf{AG}\big(\text{request} \Rightarrow \mathsf{AX}(\text{work } \mathsf{AW} \text{ grant})\big)$

DMTS

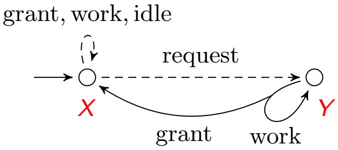


- DMTS: $\mathcal{D} = \big(S,\ S^0 \subseteq S,\ \dashrightarrow \subseteq S \times \Sigma \times S,\ \longrightarrow \subseteq S \times 2^{\Sigma \times S}\big)$
  - multiple initial states
- $\dashrightarrow$: may-transitions: behavior which is allowed
- $\longrightarrow$: disjunctive must-transitions: behavior which is required
  - $s \longrightarrow N = \{(a_1, t_1), \dots, (a_n, t_n)\}$ means: you must implement one of the behaviors $(a_1, t_1), \dots, (a_n, t_n)$
  - required $\Longrightarrow$ allowed: $\forall s \longrightarrow N : \forall (a, t) \in N : s \overset{a}{\dashrightarrow} t$
- a behavioral specification formalism

DMTS vs. ν-calculus

Direct translation between DMTS and the modal ν-calculus

- (or Hennessy-Milner logic with maximal fixed points)

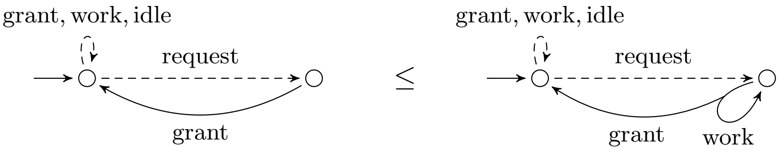


$$X = [\text{grant}, \text{idle}, \text{work}]X \wedge [\text{request}]Y$$
$$Y = (\langle\text{work}\rangle Y \vee \langle\text{grant}\rangle X) \wedge [\text{idle}, \text{request}]\mathbf{ff}$$

Refinement

A modal refinement "$\leq$" between DMTS $(S_1, S_1^0, \dashrightarrow_1, \longrightarrow_1)$ and $(S_2, S_2^0, \dashrightarrow_2, \longrightarrow_2)$:

- a relation $R \subseteq S_1 \times S_2$ such that for all $(s_1, s_2) \in R$:
- $\forall s_1 \overset{a}{\dashrightarrow} t_1 : \exists\, s_2 \overset{a}{\dashrightarrow} t_2 : (t_1, t_2) \in R$ and
- $\forall s_2 \longrightarrow N_2 : \exists\, s_1 \longrightarrow N_1 : \forall (a, t_1) \in N_1 : \exists (a, t_2) \in N_2 : (t_1, t_2) \in R$
- and $\forall s_1^0 \in S_1^0 : \exists\, s_2^0 \in S_2^0 : (s_1^0, s_2^0) \in R$



$$\mathsf{AG}\big(\text{request} \Rightarrow \text{grant}\big) \quad \leq \quad \mathsf{AG}\big(\text{request} \Rightarrow \mathsf{AX}(\text{work } \mathsf{AW} \text{ grant})\big)$$

Implementations

- implementations: standard labeled transition systems
  $S,\ s^0 \in S,\ \longrightarrow \subseteq S \times \Sigma \times S$
  - single initial state
- LTS $\subseteq$ DMTS !
- Theorem: refinement is satisfaction: $\mathcal{I} \le \mathcal{D}$ iff $\mathcal{I} \models \text{dmts2nu}(\mathcal{D})$
- implementation semantics: $[\![\mathcal{D}]\!] = \{\mathcal{I} \le \mathcal{D} \mid \mathcal{I} \text{ implementation}\}$
- Theorem: $\mathcal{D}_1 \le \mathcal{D}_2$ implies $[\![\mathcal{D}_1]\!] \subseteq [\![\mathcal{D}_2]\!]$ – sound but not complete

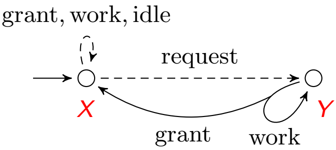
Logical operations

- Disjunction: disjoint union
  - $\mathcal{D}_1 \vee \mathcal{D}_2 = (S_1 \cup S_2, S_1^0 \cup S_2^0, \dashrightarrow_1 \cup \dashrightarrow_2, \longrightarrow_1 \cup \longrightarrow_2)$
- Conjunction: (kind of) synchronized product
  - $\mathcal{D}_1 \wedge \mathcal{D}_2 = (S_1 \times S_2, S_1^0 \times S_2^0, \dashrightarrow, \longrightarrow)$ with
  - $(s_1, s_2) \overset{a}{\dashrightarrow} (t_1, t_2)$ iff $s_1 \overset{a}{\dashrightarrow}_1 t_1$ and $s_2 \overset{a}{\dashrightarrow}_2 t_2$,
  - for all $s_1 \longrightarrow N_1$,
    $(s_1, s_2) \longrightarrow \{(a, (t_1, t_2)) \mid (a, t_1) \in N_1, (s_1, s_2) \overset{a}{\dashrightarrow} (t_1, t_2)\}$,
  - for all $s_2 \longrightarrow N_2$,
    $(s_1, s_2) \longrightarrow \{(a, (t_1, t_2)) \mid (a, t_2) \in N_2, (s_1, s_2) \overset{a}{\dashrightarrow} (t_1, t_2)\}$.
- Theorem: disjunction is least upper bound; conjunction is greatest lower bound
- Theorem: up to modal equivalence "$\equiv$", we have a bounded distributive lattice
  - $\mathcal{D}_1 \equiv \mathcal{D}_2$ iff $\mathcal{D}_1 \le \mathcal{D}_2$ and $\mathcal{D}_2 \le \mathcal{D}_1$
  - ($\le$ is *not* a partial order!)

Structural composition

- Idea: enumerate all transition possibilities
- For a DMTS $\mathcal{D} = (S, S^0, \dashrightarrow, \longrightarrow)$ and $s \in S$, let

$$\mathrm{Tran}(s) = \{M \subseteq \Sigma \times S \mid \forall (a, t) \in M : s \overset{a}{\dashrightarrow} t, \\ \forall s \longrightarrow N : N \cap M \neq \emptyset\} \subseteq 2^{\Sigma \times S}$$



$$\mathrm{Tran}(X) = \{\emptyset, \{(\mathrm{grant}, X)\}, \{(\mathrm{work}, X)\}, \{(\mathrm{idle}, X)\}, \{(\mathrm{request}, Y)\}, \\ \{(\mathrm{grant}, X), (\mathrm{work}, X)\}, \{(\mathrm{grant}, X), (\mathrm{idle}, X)\}, \dots\}$$

$$\mathrm{Tran}(Y) = \{\{(\mathrm{grant}, X)\}, \{(\mathrm{work}, Y)\}, \{(\mathrm{grant}, X), (\mathrm{work}, Y)\}\}$$

- "Acceptance automata"; special case of Walukiewicz' μ-automata

Structural composition, contd.

- $\mathcal{D}_1 \| \mathcal{D}_2 = (S_1 \times S_2, S_1^0 \times S_2^0, \mathrm{Tran})$ with
- $\mathrm{Tran}((s_1, s_2)) = \{M_1 \| M_2 \mid M_1 \in \mathrm{Tran}_1(s_1), M_2 \in \mathrm{Tran}_2(s_2)\}$, where
- $M_1 \| M_2 = \{(a, (t_1, t_2)) \mid (a, t_1) \in M_1, (a, t_2) \in M_2\}$
- Back-translation from Tran (acceptance automaton) to DMTS may give exponential blow-up
- Theorem (independent implementability):
  $\mathcal{D}_1 \le \mathcal{D}_3$ and $\mathcal{D}_2 \le \mathcal{D}_4$ imply $\mathcal{D}_1 \| \mathcal{D}_2 \le \mathcal{D}_3 \| \mathcal{D}_4$
- Hence $[\![\mathcal{D}_1]\!] \| [\![\mathcal{D}_2]\!] \subseteq [\![\mathcal{D}_1 \| \mathcal{D}_2]\!]$ – sound but not complete

Quotient

- $\mathcal{D}_1 / \mathcal{D}_2 = \big(2^{S_1 \times S_2}, \{ \{ (s_1^0, s_2^0) \mid s_1^0 \in S_1^0, s_2^0 \in S_2^0 \} \}, \mathrm{Tran} \big)$,
- with Tran too complicated to explain here...
- double exponential blow-up!
- Theorem: $\mathcal{D}_1 \| \mathcal{D}_2 \le \mathcal{D}$ iff $\mathcal{D}_2 \le \mathcal{D} / \mathcal{D}_1$
- Hence $\mathcal{I}_1 \in [\![\mathcal{D}_1]\!]$ and $\mathcal{I}_2 \in [\![\mathcal{D} / \mathcal{D}_1]\!]$ imply $\mathcal{I}_1 \| \mathcal{I}_2 \in [\![\mathcal{D}]\!]$

Residuated lattice of specifications

- Have seen already: With $\wedge$ and $\vee$, DMTS form a bounded distributive lattice up to $\equiv$
- With $\wedge$, $\vee$, $\|$ and $/$, DMTS form a bounded commutative residuated lattice up to $\equiv$:
  - $(\text{DMTS}, \|, \mathrm{U})$ is a commutative monoid (up to $\equiv$)
  - with unit $\mathrm{U} = \big(\{u\}, \{u\}, \{u \xrightarrow{a} u \mid a \in \Sigma\}\big)$ (up to $\equiv$),
  - $(\text{DMTS}, \wedge, \vee)$ is a bounded lattice (up to $\equiv$), and
  - $/$ is the residual to $\|$: $\mathcal{D}_1 \| \mathcal{D}_2 \leq \mathcal{D}$ iff $\mathcal{D}_2 \leq \mathcal{D} / \mathcal{D}_1$
- Relation to linear logic, Girard quantales

Algebraic consequences

$$
\begin{aligned}
\mathcal{D}_1 \| (\mathcal{D}_2 \vee \mathcal{D}_3) &\equiv \mathcal{D}_1 \| \mathcal{D}_2 \vee \mathcal{D}_1 \| \mathcal{D}_3 \\
(\mathcal{D}_1 \wedge \mathcal{D}_2) / \mathcal{D}_3 &\equiv \mathcal{D}_1 / \mathcal{D}_3 \wedge \mathcal{D}_2 / \mathcal{D}_3 \\
\mathcal{D}_1 \| (\mathcal{D}_2 / \mathcal{D}_1) &\leq \mathcal{D}_2 \\
(\mathcal{D}_1 \| \mathcal{D}_2) / \mathcal{D}_1 &\leq \mathcal{D}_2 \\
\mathcal{D} / \mathrm{U} &\equiv \mathcal{D} \\
\mathrm{U} &\leq \mathcal{D} / \mathcal{D} \\
(\mathcal{D}_1 / \mathcal{D}_2) / \mathcal{D}_3 &\equiv \mathcal{D}_1 / (\mathcal{D}_2 \| \mathcal{D}_3) \\
(\mathrm{U} / \mathcal{D}_1) \| (\mathrm{U} / \mathcal{D}_2) &\leq \mathrm{U} / (\mathcal{D}_1 \| \mathcal{D}_2)
\end{aligned}
$$