Where is the focus...

<table>
<thead>
<tr>
<th>Setting and motivation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modern (software and hardware) systems: several (orthogonal) requirements, complexity of the architecture, high repairing costs, ...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>General approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic formal modeling and analysis methods as a support to system design.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Core idea</th>
</tr>
</thead>
<tbody>
<tr>
<td>A study of interest: <strong>equivalence checking</strong> → basic ingredients, applications, approximations.</td>
</tr>
</tbody>
</table>
What are Formal Methods?

...refer to mathematically rigorous techniques and tools for the specification, design and verification of systems.
What are Formal Methods?

... refer to mathematically rigorous techniques and tools for the specification, design and verification of systems.

“mathematically rigorous” means that the specifications used in formal methods are well-formed statements (e.g. in a mathematical logic) and that the formal verifications are rigorous (e.g. deductions in that logic) and can be checked by a mechanical process.
What are Formal Methods?

... refer to mathematically rigorous techniques and tools for the specification, design and verification of systems.

... they provide means to symbolically study the entire state space of a design (whether hardware or software) and establish properties that, for instance, are true for all possible inputs.
Using Formal Methods

Evolution
Sequential programming, parallel computing, concurrent and distributed systems, service-oriented computing, cloud computing, ... 

Problem
Enormous complexity of real systems.
Several approaches are used to overcome the astronomically-sized state spaces associated with real complex systems:

- Apply formal methods to high-level designs where most of the details are abstracted away.
- Verify partial views of the system.
- Analyze models of software and hardware where variables are discretized and ranges drastically reduced.

Does THE ONE exist?

Each application domain requires different modeling methods and different proof approaches. Furthermore, even within a particular application domain, different phases of the life-cycle may be best served by different tools and techniques.
Using Formal Methods

**Models**
- Logic calculi
- Formal languages
- Automata theory
- Program semantics
- Algebraic data types and type systems
- ...

**Techniques**
- Model checking
- **Equivalence checking**
- Type checking
- Static analysis
- ...

Behavioral Equivalences

Do syntactically different programs/models/systems exhibit the same behavior from the viewpoint of an external observer?

Different Perspectives

- Yes, whenever they can execute the same sequences of events/instructions.
- Yes, whenever under the same tests they react in the same way.
- Yes, whenever they are able to mimic each other’s behavior step by step.
An Example: Bisimulation Equivalence

1. Define formally labelled transition systems. 
\[ \text{LTS} = (S, \text{Act}, \rightarrow) \] where:
   - \( S \) is a set of states.
   - \( \text{Act} \) is a set of actions.
   - \( \rightarrow \subseteq S \times \text{Act} \times S \) is a transition relation.

Use \( s \xrightarrow{\alpha} s' \) to denote \( (s, \alpha, s') \in \rightarrow \).
2. Define formally a behavioral equivalence.
A binary relation $R$ on $S \times S$ is a strong bisimulation whenever for $(s, t) \in R$:

- if $s \xrightarrow{\alpha} s'$ then there exists $t' \in S$ such that $t \xrightarrow{\alpha} t'$ and $(s', t') \in R$.
- if $t \xrightarrow{\alpha} t'$ then there exists $s' \in S$ such that $s \xrightarrow{\alpha} s'$ and $(s', t') \in R$. 

Are they equivalent?
An Example: Bisimulation Equivalence

3. $s$ is strongly bisimilar to $t$, $s \sim t$, if there exists a strong bisimulation $R$ such that $(s, t) \in R$ ($\sim$ is an equivalence).

$s_0 \sim t_0$?
3. Yes, the relation is $R = \{(s_0, t_0), (s_1, t_1), (s_2, t_1), (s_1, s_2), (s_3, t_2)\}$ and the related partition contains the classes $\{s_0, t_0\}$, $\{s_1, s_2, t_1\}$, and $\{s_3, t_2\}$. 

Are they equivalent?
An Example: Bisimulation Equivalence

Are they strongly bisimilar?
An Example: Bisimulation Equivalence

Are they strongly bisimilar? Yes!
An Example: Bisimulation Equivalence

Are they strongly probabilistic bisimilar? Yes!
An Example: Bisimulation Equivalence

Are they weakly bisimilar? Yes!
An Example: Bisimulation Equivalence

Are they weakly probabilistic bisimilar? Yes, but . . .
Equivalence Checking

Applications

- Relating a process model to a reference model.
- Verifying substitutions/transformations/reductions that are expected to preserve system properties.
- Noninterference analysis.
Noninterference Theory

Original Idea

- A group of high-security level users, employing confidential operations only, is not interfering with a group of low-security level users, observing public operations only, if what the first group of users can do with the confidential operations has no effect on what the second group of users can see.

- In the security setting, noninterference analysis can reveal direct and indirect information flows, called *covert channels*, that violate the access policies based on the different access clearances assigned to different user groups.
Noninterference Theory

### Original Idea

- A group of high-security level users, employing confidential operations only, is not interfering with a group of low-security level users, observing public operations only, if what the first group of users can do with the confidential operations has no effect on what the second group of users can see.

- In the security setting, noninterference analysis can reveal direct and indirect information flows, called *covert channels*, that violate the access policies based on the different access clearances assigned to different user groups.
Noninterference Theory

Implementation

- Action names (events) are divided into two disjoint sets:
  - *High*, representing system activities at high-security level
  - *Low*, representing system activities at low-security level

- A system model $Q$ has no covert channels if the system view where all the high-level activities are *hidden* to low-level observers, is indistinguishable with respect to the system view where these activities are *prevented* from execution.
Noninterference Theory

General Idea

- A system execution can be viewed as an information flow.
- A group of system components (high components), described by a certain set of behaviors, is not interfering with another group of system components (low components) if the behaviors of the first group of components have no effect on what the second group of components can see.
- In this more general setting, noninterference analysis can reveal covert channels indicating the existence of undesired information flows among component behaviors that are responsible for compromising different aspects.
Noninterference Theory

General Idea

- A system execution can be viewed as an information flow.
- A group of system components (high components), described by a certain set of behaviors, is not interfering with another group of system components (low components) if the behaviors of the first group of components have no effect on what the second group of components can see.
- In this more general setting, noninterference analysis can reveal covert channels indicating the existence of undesired information flows among component behaviors that are responsible for compromising different aspects.
Noninterference Theory

General Idea

- A system execution can be viewed as an information flow.
- A group of system components (high components), described by a certain set of behaviors, is not interfering with another group of system components (low components) if the behaviors of the first group of components have no effect on what the second group of components can see.
- In this more general setting, noninterference analysis can reveal covert channels indicating the existence of undesired information flows among component behaviors that are responsible for compromising different aspects.
Noninterference: granularity of information

Multilevel secure computing:
*Low vs. High*

- Information flow detection:
  - Deterministic/Nondeterministic
  - Noninterference
- Covert channel capacity:
  - Quantitative
  - Noninterference

- exact analysis
- 0/1 result

Nondeducibility (Sutherland), Causality (Roscoe)
Nondeducibility on Strategies (Wittbold&Johnson)
Generalized Noninterference, Restrictiveness
  (McCullough, McLean)
Nondeducibility on Compositions (Focardi&Gorrieri)

- approximate analysis
- Probabilistic noninterference:
  - [0; 1] result
  - best-case/average/worst-case

Gray, Di Pierro, Aldini, Smith, Sabelfeld, ... 
Aldini&Bernardo 

- Stochastic noninterference
Why Approximate Equivalence Checking

¬ Perfect equivalence

⇓

Quantitative comparison

⇓

Numbers!
Approximating equivalence

Most common approaches consider bisimulation

- It has a suitable modal logic characterization.
- It can be relaxed by turning the equivalence into a relation.
Pseudometric Approach

[Desharnais et al., vBW, ...]

- Logical characterization of bisimulation: $\mathcal{L} := \top \mid \phi_1 \land \phi_2 \mid \langle a \rangle_q \phi$
- From the logic-based characterization to the functional expressions based characterization:
  
  $$f := 1 \mid 1 - f \mid \langle a \rangle f \mid \min(f_1, f_2) \mid \sup_{i \in \mathbb{N}} f_i \mid f \ominus q$$
Pseudometric Approach

[Desharnais et al., vBW, ...]

- Logical characterization of bisimulation: \( \mathcal{L} := \top \mid \phi_1 \wedge \phi_2 \mid \langle a \rangle q \phi \)

- From the logic-based characterization to the functional expressions based characterization:
  \[
  f := 1 \mid 1 - f \mid \langle a \rangle f \mid \min(f_1, f_2) \mid \sup_{i \in \mathbb{N}} f_i \mid f \ominus q
  \]
  
  \* \( 1(s) = 1 \) stands for \( \top \)
  
  \* \( (1 - f)(s) = 1 - f(s) \)
  
  \* \( \langle a \rangle f(s) = c \int_{S} f(t) \tau_a(s, dt) \) stands for prefixing
  
  \* \( (f \ominus q)(s) = \max(f(s) - q, 0) \) stands for \( > p \)
Pseudometric Approach

Logical characterization of bisimulation: \( \mathcal{L} := \top | \phi_1 \land \phi_2 | \langle a \rangle q \phi \)

From the logic-based characterization to the functional expressions based characterization:
\[
f := 1 | 1 - f | \langle a \rangle f | \min(f_1, f_2) | \sup_{i \in \mathbb{N}} f_i | f \ominus q
\]

\( s \) and \( s' \) are bisimilar iff they satisfy the same logical formulas iff they have the same values for each functional expression.

Pseudometric: \( d^c(P, Q) = \sup_{f \in \mathcal{F}} | f_P(p_0) - f_Q(q_0) | \)
An Example

⟨a⟩⟨a⟩1 evaluates to $3c^2/4$ at state $s_0$ and to 0 elsewhere

⟨a⟩(⟨a⟩1 ⊖ c/2) evaluates to $3c^2/8$ at state $s'_0$

⟨a⟩(⟨a⟩1 ⊖ c/2) evaluates to $c^2/4$ at state $s'_0$
Pseudometric Approach

\[ d^c(P, Q) = \sup_{f \in \mathcal{F}^c} |f_P(p_0) - f_Q(q_0)| \]

Limitations concerning the interpretation of the distance

- it is state-based, what about an activity-oriented setting...
- any pair of states can be considered, which comparisons make sense...
An Example

if $c = 1$ then $s_3$ ($s'_3$) is as important as $s_0$ ($s'_0$)

no functional expression reveals that the probability of reaching $s_3$ ($s'_3$) is 1
An Example

An approximating relation should:

- evaluate the distance between states in the same group – obviously \((s_0, s'_0) \in R\),
- employ a notion of distance taking into account the probability of being at the states under comparison.
A relation $R \subseteq S \times S$ is a:

1. probabilistic bisimulation with $\varepsilon$ precision if whenever $(s, s') \in R$, then for all $C$ in the partition induced by $R$ and $\forall a \in \text{Act. } d(s, s', a, C) \leq \varepsilon$. [ADiP,Ald]

2. $\varepsilon$-simulation if whenever $sRt$, then $\forall a \in \text{Act}, X \subseteq S. h_a(t, R(X)) \geq h_a(s, X) - \varepsilon$. Then, $R$ is a $\varepsilon$-bisimulation if it is symmetric and a $\varepsilon$-simulation. [Desh. et al.]

3. $\varepsilon$-bisimulation if whenever $sRt$, then the norm of a linear operator applied to the matrix representations of $s$ and $t$ with respect to a $R$-based classification operator is confined by $\varepsilon$. [DiPHW]
Other Approaches: Approx. Bisimulation

1. Has a clear numerical interpretation (relation with quasi-lumpability), but not a poly-time verification algorithm.
2. Has logic-based and game-theoretic characterizations, a poly-time verification algorithm, but strong usability limitations.
3. Is efficient, but the measure strictly depends on the chosen norms and classification linear operators.
Probabilistic Bisimulation with $\varepsilon$ Precision

- $d^R_A(s, s', a, C) = w(s, s') \cdot | \text{Prob}(s, a, C) - \text{Prob}(s', a, C) |$
- $\delta^R = \max \{ | d^R_A(s, s', a, C) | \mid s \in S_1, s' \in S_2, (s, s') \in R, a \in \text{Act}, C \in S_1 \cup S_2/R \}$

- Computing $\delta^R$ is an easy task.
- Computing $\inf_R \delta^R$ requires a poly-time algorithm to determine a local optimum, while meta-heuristic techniques can be applied to the searching of the closest approximation of bisimulation.
A Different Approach

- ...based on testing equivalence.
- ...dealing with temporal and probabilistic aspects of the observed behaviors.
- ...including a quantitative comparison of the observed behaviors based on typical behaviors.
Testing Similarity

Introduces three tolerance thresholds:

- **temporal threshold** $t$: one system is faster/slower than the other (up to $t$).
- **probabilistic threshold** $p$: the two systems react with different probabilities (up to $p$).
- **precision** $p$ and **recall** $r$: similar reactions are observed under similar tests, where test similarity is measured through fitness functions called precision and recall.

The resulting approximation is a conservative extension of the related equivalence, satisfies the triangular inequality, and is checkable in poly-time.
Noninterference: granularity of information

Multilevel secure computing:
  Low vs. High

Information flow detection:
  Deterministic/Nondeterministic Noninterference

Covert channel capacity:
  Quantitative Noninterference

- **approximate** analysis
- Probabilistic noninterference:
  [0; 1] result
- Stochastic noninterference

- exact analysis
- 0/1 result
Objective

1. Models
   ↓
2. Semantics
   ↓
3. Metric
   ↓
4. Feedback
   ↓
5. Optimization


References