A Fault Tolerance Bisimulation Proof for Consensus

Adrian Francalanza[1] Matthew Hennessy[2]

[1]Department of Computing
Imperial College

[2]Department of Informatics
University of Sussex

European Symposium on Programming, 2007

Outline

Outline

Distributed Systems: Consensus

Consensus Setting

- n autonomous participants who may independently fail
- hold a value $v \in V$.
- must decide on a value $v' \in V$.

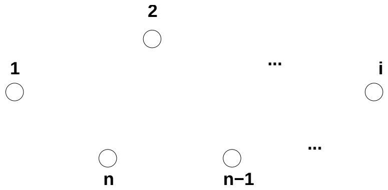
Defining Correctness of Consensus

Termination: All non-failing participants must eventually decide.

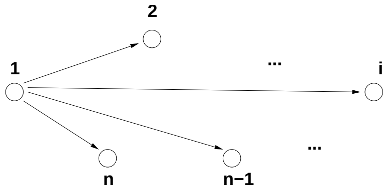
Agreement: No two participants decide on different values.

Validity: If all participants are given the same value $v \in V$ as input, then v is the only possible decision value.

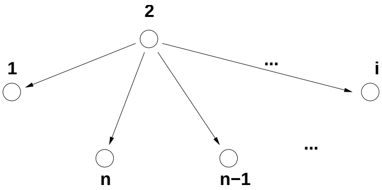
Rotating Coordinator Algorithm



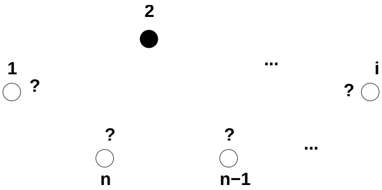
Rotating Coordinator Algorithm



Rotating Coordinator Algorithm



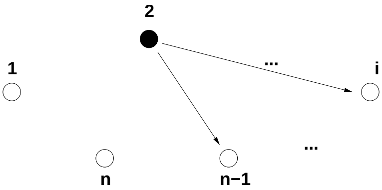
Rotating Coordinator Algorithm



Problems caused by Failure

Decision Blocking: a participant waiting forever for a value to be broadcast from a crashed co-ordinator

Rotating Coordinator Algorithm



Problems caused by Failure

Decision Blocking: a participant waiting forever for a value to be broadcast from a crashed co-ordinator

Corrupted Broadcast: a co-ordinator broadcasts its values to a *subset* of the participants before crashing.

Distributed Systems: Consensus with Perfect Failure Detection

Rotating Coordinator Algorithm for Participant *i*

```
part[1..n]; \\ array of n participants
x_i := input;  \\ initialise
for r := 1 to n do {
   if (r = i) then broadcast(x_i);
   if alive(part[r]) then x_i:= input(part[r]);
}
output x_i;  \\ decide
```

Distributed Systems: Consensus with Perfect Failure Detection

Rotating Coordinator Algorithm for Participant i

```
part[1..n]; \\ array of n participants
x_i := input;  \\ initialise
for r := 1 to n do {
   if (r = i) then broadcast(x_i);
   if alive(part[r]) then x_i:= input(part[r]);
}
output x_i;  \\ decide
```

Distributed Systems: Consensus with Perfect Failure Detection

Rotating Coordinator Algorithm for Participant *i*

```
part[1..n]; \\ array of n participants
x_i := input;  \\ initialise
for r := 1 to n do {
   if (r = i) then broadcast(x_i);
   if alive(part[r]) then x_i:= input(part[r]);
}
output x_i;  \\ decide
```

Distributed Systems: Consensus with Perfect Failure Detection

Rotating Coordinator Algorithm for Participant *i*

```
part[1..n]; \\ array of n participants
x_i := input;  \\ initialise
for r := 1 to n do {
   if (r = i) then broadcast(x_i);
   if alive(part[r]) then x_i:= input(part[r]);
}
output x_i;  \\ decide
```

The Process Calculi Way

Language: Parallel composition, atomic actions, action hiding, reduction semantics

Bisimulation: lts and $\approx$ characterises behavioural equivalence

The Process Calculi Way

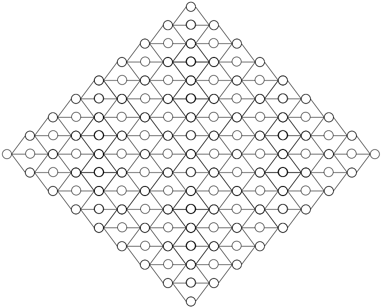
Language: Parallel composition, atomic actions, action hiding, reduction semantics

Bisimulation: lts and $\approx$ characterises behavioural equivalence

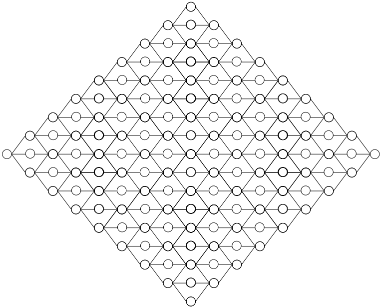
Theorem

$$P \,|\, P \,|\, \ldots \,|\, P \;\approx\; SPEC$$

The Process Calculi Way (2)



The Process Calculi Way (2)



- complex to understand
- hard to digest
- not intuitive

The Process Calculi Way ... with Failures!

Language: Parallel composition, atomic actions, action hiding, reduction semantics with failures.

Bisimulation: lts and $\approx$ characterises behavioural equivalence

Theorem

$$l_1[\![P]\!] \mid \ldots \mid l_n[\![P]\!] \quad \approx \quad SPEC\_FAIL$$

The Process Calculi Way ... with Failures!

Language: Parallel composition, atomic actions, action hiding, reduction semantics with failures.

Bisimulation: lts and ≈ characterises behavioural equivalence

Theorem

$$l_1[\![P]\!] \mid \ldots \mid l_n[\![P]\!] \quad \approx \quad SPEC\_FAIL$$

The Process Calculi Way ... with Failures!

Language: Parallel composition, atomic actions, action hiding, reduction semantics with failures.

$$\frac{}{(\Gamma, n+1) \triangleright M \xrightarrow{\tau} (\Gamma - l, n) \triangleright M} \; \Gamma \vdash l : \textbf{alive}$$

Bisimulation: lts and $\approx$ characterises behavioural equivalence

Theorem

$$\Gamma, n-1 \triangleright l_1[\![P]\!] \,|\, \ldots \,|\, l_n[\![P]\!] \;\approx\; \Gamma, n-1 \triangleright SPEC\_FAIL$$

The Process Calculi Way ... with Failures!

Expressing Participant i in our process calculus

(Participant)

$$\mathrm{P}^x_{i,r} \stackrel{\text{def}}{=} \mathrm{R}^x_{i,r} \mid \mathrm{B}^x_{i,r} \qquad x \in \{true, false\},\; r \leq n$$

$$\mathrm{P}^x_{i,n+1} \stackrel{\text{def}}{=} \overline{dec^x_i} \qquad x \in \{true, false\}$$

(Recieve)

$$\mathrm{R}^x_{i,r} \stackrel{\text{def}}{=} true_{i,r}.\mathrm{P}^{true}_{i,r+1} + false_{i,r}.\mathrm{P}^{false}_{i,r+1} + \mathrm{susp}\ l_r.\mathrm{P}^x_{i,r+1}$$

(Broadcast)

$$\mathrm{B}^x_{i,i} \stackrel{\text{def}}{=} \prod_{j=1}^{n} \overline{x_{j,r}} \qquad x \in \{true, false\}$$

$$\mathrm{B}^x_{i,r} \stackrel{\text{def}}{=} \mathbf{0} \qquad x \in \{true, false\},\; r \neq i$$

The Process Calculi Way ... with Failures!

Expressing Participant i in our process calculus

(Participant)

$$P^{x}_{i,r} \stackrel{\text{def}}{=} R^{x}_{i,r} \mid B^{x}_{i,r} \qquad x \in \{true, false\},\ r \leq n$$

$$P^{x}_{i,n+1} \stackrel{\text{def}}{=} \overline{dec^{x}_{i}} \qquad x \in \{true, false\}$$

(Recieve)

$$R^{x}_{i,r} \stackrel{\text{def}}{=} true_{i,r}.P^{true}_{i,r+1} + false_{i,r}.P^{false}_{i,r+1} + \text{susp}\ l_r.P^{x}_{i,r+1}$$

(Broadcast)

$$B^{x}_{i,i} \stackrel{\text{def}}{=} \prod_{j=1}^{n} \overline{x_{j,r}} \qquad x \in \{true, false\}$$

$$B^{x}_{i,r} \stackrel{\text{def}}{=} \mathbf{0} \qquad x \in \{true, false\},\ r \neq i$$

The Process Calculi Way ... with Failures!

Expressing Participant i in our process calculus

(Participant)

$$\mathsf{P}^x_{i,r} \stackrel{\text{def}}{=} \mathsf{R}^x_{i,r} \mid \mathsf{B}^x_{i,r} \qquad x \in \{\mathit{true}, \mathit{false}\},\ r \leq n$$
$$\mathsf{P}^x_{i,n+1} \stackrel{\text{def}}{=} \overline{\mathit{dec}^x_i} \qquad x \in \{\mathit{true}, \mathit{false}\}$$

(Recieve)

$$\mathsf{R}^x_{i,r} \stackrel{\text{def}}{=} \mathit{true}_{i,r}.\mathsf{P}^{\mathit{true}}_{i,r+1} + \mathit{false}_{i,r}.\mathsf{P}^{\mathit{false}}_{i,r+1} + \mathrm{susp}\ l_r.\mathsf{P}^x_{i,r+1}$$

(Broadcast)

$$\mathsf{B}^x_{i,i} \stackrel{\text{def}}{=} \prod_{j=1}^{n} \overline{x_{j,r}} \qquad x \in \{\mathit{true}, \mathit{false}\}$$
$$\mathsf{B}^x_{i,r} \stackrel{\text{def}}{=} \mathbf{0} \qquad x \in \{\mathit{true}, \mathit{false}\},\ r \neq i$$

The Process Calculi Way ... with Failures!

Expressing Participant i in our process calculus

(Participant)

$$\mathsf{P}^{x}_{i,r} \stackrel{\text{def}}{=} \mathsf{R}^{x}_{i,r} \mid \mathsf{B}^{x}_{i,r} \qquad x \in \{true, false\},\ r \leq n$$

$$\mathsf{P}^{x}_{i,n+1} \stackrel{\text{def}}{=} \overline{dec^{x}_{i}} \qquad x \in \{true, false\}$$

(Recieve)

$$\mathsf{R}^{x}_{i,r} \stackrel{\text{def}}{=} true_{i,r}.\mathsf{P}^{true}_{i,r+1} + false_{i,r}.\mathsf{P}^{false}_{i,r+1} + \mathsf{susp}\ l_r.\mathsf{P}^{x}_{i,r+1}$$

(Broadcast)

$$\mathsf{B}^{x}_{i,i} \stackrel{\text{def}}{=} \prod_{j=1}^{n} \overline{x_{j,r}} \qquad x \in \{true, false\}$$

$$\mathsf{B}^{x}_{i,r} \stackrel{\text{def}}{=} \mathbf{0} \qquad x \in \{true, false\},\ r \neq i$$

The Process Calculi Way ... with Failures!

Expressing Participant i in our process calculus

(Participant)

$$\mathsf{P}^x_{i,r} \stackrel{\text{def}}{=} \mathsf{R}^x_{i,r} \mid \mathsf{B}^x_{i,r} \qquad x \in \{\mathit{true}, \mathit{false}\},\ r \leq n$$

$$\mathsf{P}^x_{i,n+1} \stackrel{\text{def}}{=} \overline{\mathit{dec}^x_i} \qquad x \in \{\mathit{true}, \mathit{false}\}$$

(Recieve)

$$\mathsf{R}^x_{i,r} \stackrel{\text{def}}{=} \mathit{true}_{i,r}.\mathsf{P}^{\mathit{true}}_{i,r+1} + \mathit{false}_{i,r}.\mathsf{P}^{\mathit{false}}_{i,r+1} + \text{susp } l_r.\mathsf{P}^x_{i,r+1}$$

(Broadcast)

$$\mathsf{B}^x_{i,i} \stackrel{\text{def}}{=} \prod_{j=1}^{n} \overline{x_{j,r}} \qquad x \in \{\mathit{true}, \mathit{false}\}$$

$$\mathsf{B}^x_{i,r} \stackrel{\text{def}}{=} \mathbf{0} \qquad x \in \{\mathit{true}, \mathit{false}\},\ r \neq i$$

The Process Calculi Way ... with Failures!

Expressing Participant i in our process calculus

(Participant)

$$\mathsf{P}^x_{i,r} \stackrel{\text{def}}{=} \mathsf{R}^x_{i,r} \mid \mathsf{B}^x_{i,r} \qquad x \in \{\mathit{true}, \mathit{false}\},\ r \leq n$$

$$\mathsf{P}^x_{i,n+1} \stackrel{\text{def}}{=} \overline{dec^x_i} \qquad x \in \{\mathit{true}, \mathit{false}\}$$

(Recieve)

$$\mathsf{R}^x_{i,r} \stackrel{\text{def}}{=} \mathit{true}_{i,r}.\mathsf{P}^{\mathit{true}}_{i,r+1} + \mathit{false}_{i,r}.\mathsf{P}^{\mathit{false}}_{i,r+1} + \text{susp } l_r.\mathsf{P}^x_{i,r+1}$$

(Broadcast)

$$\mathsf{B}^x_{i,i} \stackrel{\text{def}}{=} \prod_{j=1}^{n} \overline{x_{j,r}} \qquad x \in \{\mathit{true}, \mathit{false}\}$$

$$\mathsf{B}^x_{i,r} \stackrel{\text{def}}{=} \mathbf{0} \qquad x \in \{\mathit{true}, \mathit{false}\},\ r \neq i$$

The Process Calculi Way ... with Failures!

Expressing Participant i in our process calculus

(Participant)

$$\mathsf{P}^{x}_{i,r} \stackrel{\text{def}}{=} \mathsf{R}^{x}_{i,r} \mid \mathsf{B}^{x}_{i,r} \qquad x \in \{true, false\},\ r \leq n$$

$$\mathsf{P}^{x}_{i,n+1} \stackrel{\text{def}}{=} \overline{dec^{x}_{i}} \qquad x \in \{true, false\}$$

(Recieve)

$$\mathsf{R}^{x}_{i,r} \stackrel{\text{def}}{=} true_{i,r}.\mathsf{P}^{true}_{i,r+1} + false_{i,r}.\mathsf{P}^{false}_{i,r+1} + \mathsf{susp}\ l_r.\mathsf{P}^{x}_{i,r+1}$$

(Broadcast)

$$\mathsf{B}^{x}_{i,i} \stackrel{\text{def}}{=} \prod_{j=1}^{n} \overline{x_{j,r}} \qquad x \in \{true, false\}$$

$$\mathsf{B}^{x}_{i,r} \stackrel{\text{def}}{=} \mathbf{0} \qquad x \in \{true, false\},\ r \neq i$$

The Process Calculi Way ... with Failures!

Composing n Participants to solve Consensus

$$\mathsf{C} \stackrel{\text{def}}{=} \left(\nu^{n}_{i,r=1}\ \begin{array}{l} true_{i,r}, \\ false_{i,r} \end{array}\right) \prod^{n}_{i=1} l_i \left[\!\!\left[\begin{array}{l} prop_i^{true}.\mathsf{P}^{true}_{i,1} \\ +prop_i^{false}.\mathsf{P}^{false}_{i,1} \end{array} \right]\!\!\right]$$

The Process Calculi Way ... with Failures!

Composing n Participants to solve Consensus

$$\mathsf{C} \stackrel{\text{def}}{=} \left(\nu^{n}_{i,r=1} \begin{array}{l} true_{i,r}, \\ false_{i,r} \end{array}\right) \prod^{n}_{i=1} l_i \left[\!\!\left[\begin{array}{l} prop_i^{true}.\mathsf{P}_{i,1}^{true} \\ +prop_i^{false}.\mathsf{P}_{i,1}^{false} \end{array} \right]\!\!\right]$$

The Process Calculi Way ... with Failures!

Composing n Participants to solve Consensus

$$\mathsf{C} \stackrel{\text{def}}{=} \left(\nu^{n}_{i,r=1}\; \begin{array}{l} true_{i,r}, \\ false_{i,r} \end{array}\right) \prod^{n}_{i=1} l_i \left[\!\!\left[\begin{array}{l} prop^{true}_{i}.\mathsf{P}^{true}_{i,1} \\ +prop^{false}_{i}.\mathsf{P}^{false}_{i,1} \end{array} \right]\!\!\right]$$

The Process Calculi Way ... with Failures!

Composing n Participants to solve Consensus

$$\mathsf{C} \stackrel{\text{def}}{=} \left(\nu_{i,r=1}^{n} \begin{array}{l} true_{i,r}, \\ false_{i,r} \end{array} \right) \prod_{i=1}^{n} l_i \left[\!\left[\begin{array}{l} prop_i^{true}.\mathsf{P}_{i,1}^{true} \\ + prop_i^{false}.\mathsf{P}_{i,1}^{false} \end{array} \right]\!\right]$$

The Process Calculi Way ... with Failures!

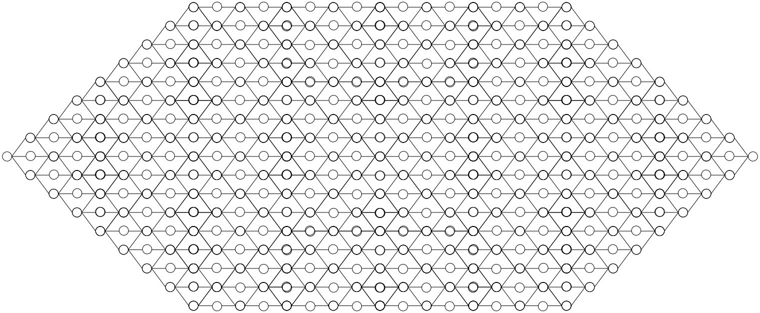
Composing n Participants to solve Consensus

$$\mathrm{C} \stackrel{\text{def}}{=} \left(\nu^{n}_{i,r=1} \begin{array}{l} true_{i,r}, \\ false_{i,r} \end{array} \right) \prod^{n}_{i=1} l_i \left[\!\left[\begin{array}{l} prop^{true}_{i}.\mathrm{P}^{true}_{i,1} \\ +prop^{false}_{i}.\mathrm{P}^{false}_{i,1} \end{array} \right]\!\right]$$

Specification in the presence of Failure

VERY COMPLEX!

Bisimulation with Failure



An even bigger, more complex bisimulation!

Outline

Methodology (1): Testing Harnesses

Language: Parallel composition, atomic actions, action hiding, reduction semantics w. failures, immortal location $\star$.

Bisimulation: lts and $\approx$ characterises behavioural equivalence w.r.t immortal observers

Theorem

$$\Gamma, n-1 \triangleright (\nu \tilde{m})(\star[\![Q]\!] \,|\, l_1[\![P]\!] \,|\, \ldots \,|\, l_n[\![P]\!]) \quad \approx \quad \Gamma, 0 \triangleright \star[\![SPEC]\!]$$

Methodology (1): Testing Harnesses

Language: ...

Bisimulation: ...

Theorem

$$\Gamma, n-1 \triangleright (\nu \tilde{m})(\star[\![Q]\!] \mid l_1[\![P]\!] \mid \ldots \mid l_n[\![P]\!]) \;\approx\; \Gamma, 0 \triangleright \star[\![SPEC]\!]$$

Advantages

1. Simplifies specification formulation
2. Permits separate tests for correctness criteria

Methodology (1): Testing Harnesses

Harnesses For Consensus

(Initialisation)

$$\mathsf{I}^{x} \stackrel{\text{def}}{=} start.\prod_{i=1}^{n} \overline{prop_i^x} \qquad x \in \{true,\ false\}$$

$$\mathsf{I}^{gen} \stackrel{\text{def}}{=} start.\prod_{i=1}^{n}(\overline{prop_i^{true}} + \overline{prop_i^{false}})$$

(Agreement)

$$\mathsf{A}_i^{x} \stackrel{\text{def}}{=} dec_i^{x}.\mathsf{A}_{i+1}^{x} + \mathsf{susp}\ l_i.\mathsf{A}_{i+1}^{x} \qquad x \in \{true,\ false\},\ i \leq n$$

$$\mathsf{A}_{n+1}^{x} \stackrel{\text{def}}{=} \overline{ok} \qquad x \in \{true,\ false\}$$

$$\mathsf{A}_i^{gen} \stackrel{\text{def}}{=} dec_i^{true}.\mathsf{A}_{i+1}^{true} + dec_i^{false}.\mathsf{A}_{i+1}^{false} + \mathsf{susp}\ l_i.\mathsf{A}_{i+1}^{gen}$$

Methodology (1): Testing Harnesses

Language: . . .
Bisimulation: . . .

Theorem

Agreement and Termination

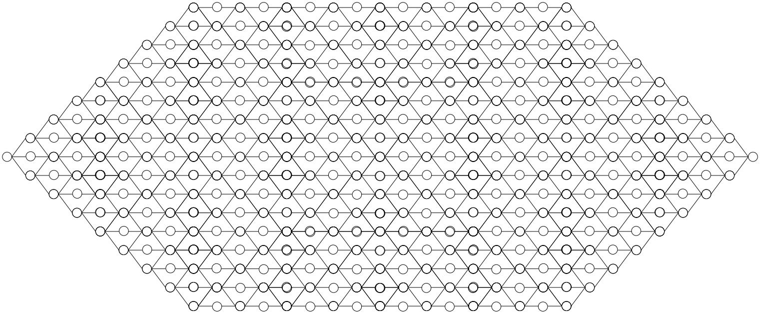
$$\Gamma, n-1 \triangleright (\nu\tilde{m})(\star[\![I^{gen} \mid A_1^{gen}]\!] \mid C) \approx \Gamma, 0 \triangleright \star[\![start.\overline{ok}]\!]$$

Validity

$$\Gamma, n-1 \triangleright (\nu\tilde{m})(\star[\![I^{true} \mid A_1^{true}]\!] \mid C) \approx \Gamma, 0 \triangleright \star[\![start.\overline{ok}]\!]$$
$$\Gamma, n-1 \triangleright (\nu\tilde{m})(\star[\![I^{false} \mid A_1^{false}]\!] \mid C) \approx \Gamma, 0 \triangleright \star[\![start.\overline{ok}]\!]$$

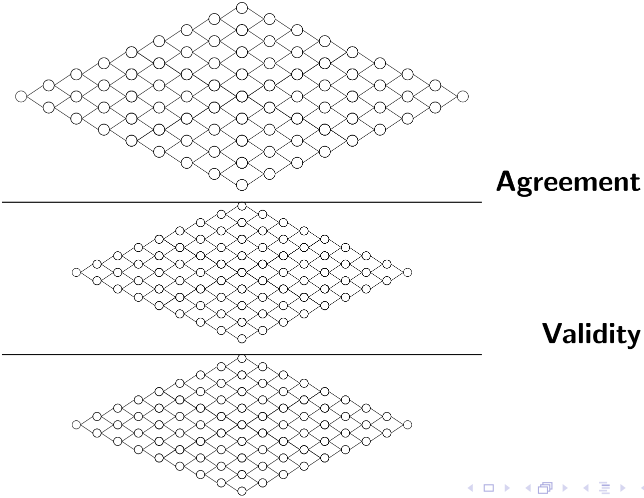
$\tilde{m} = \prod_{i=1}^{n} prop_i^{true}, prop_i^{false}, dec_i^{true}, dec_i^{false}$

Methodology (1): ... on the bisimulation front



Where we left off... (big, complex bisimulation!)

Methodology (1): ... on the bisimulation front



Methodology (2): Fault Tolerance

Language: ...

Bisimulation: ...

Theorem

$$\Gamma, n-1 \triangleright (\nu \tilde{m})(\star[\![I | A]\!] \mid C) \quad \approx \quad \Gamma, 0 \triangleright \star[\![start.\overline{ok}]\!]$$

Methodology (2): Fault Tolerance

Language: ...

Bisimulation: ...

Theorem

$$\Gamma, n-1 \triangleright (\nu \tilde{m})(\star[\![I | A]\!] \mid \underbrace{C}_{\textit{Induce failure}}) \approx \Gamma, 0 \triangleright \star[\![\mathit{start}.\overline{\mathit{ok}}]\!]$$

Methodology (2): Fault Tolerance

Language: ...

Bisimulation: ...

Theorem

$$\Gamma, n-1 \triangleright (\nu \tilde{m}) \overbrace{\star[\![I|A]\!]}^{\textit{Preserve observable behaviour}} \;|\; \underbrace{C}_{\textit{Induce failure}}) \approx \Gamma, 0 \triangleright \overbrace{\star[\![start.\overline{ok}]\!]}^{\textit{Preserve observable behaviour}}$$

Methodology (2): Fault Tolerance

Language: ...

Bisimulation: ...

Theorem

$$\Gamma, n-1 \triangleright (\nu \tilde{m})(\star[\![I|A]\!] \mid C) \quad \approx \quad \Gamma, 0 \triangleright \star[\![start.\overline{ok}]\!]$$

Methodology (2): Fault Tolerance

Language: ...

Bisimulation: ...

Theorem

Basic Correctness

$$\Gamma, 0 \triangleright (\nu \tilde{m})(\star[\![I|A]\!] \mid C) \quad \approx \quad \Gamma, 0 \triangleright \star[\![start.\overline{ok}]\!]$$

Methodology (2): Fault Tolerance

Language: . . .

Bisimulation: . . .

Theorem

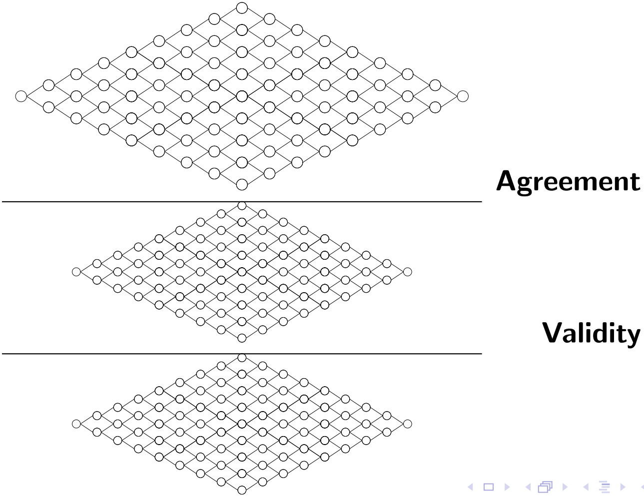
Basic Correctness

$$\Gamma, 0 \triangleright (\nu \tilde{m})(\star[\![I|A]\!] \mid C) \quad \approx \quad \Gamma, 0 \triangleright \star[\![start.\overline{ok}]\!]$$

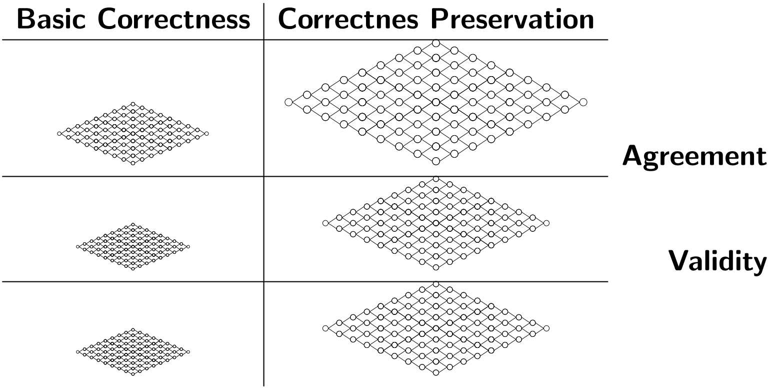
Correctness Preservation

$$\Gamma, n-1 \triangleright (\nu \tilde{m})(\star[\![I|A]\!] \mid C) \quad \approx \quad \Gamma, 0 \triangleright (\nu \tilde{m})(\star[\![I|A]\!] \mid C)$$

Methodology (2): ... on the bisimulation front



Methodology (2): ... on the bisimulation front



Methodology (2): Advantages

1. **Natural** way how to analyse algorithms in the presence of failure.
2. **Stages are Independent**:
   - Test them in parallel
   - Simpler (failure-free) stage can be used as a vetting stage
3. **Refined Up-to Techniques**: we have different confluences and structural equivalences under different failure settings.

Methodology (3): Up-to Techniques

Confluence Properties

$$\langle \Gamma, n \rangle \triangleright N \overset{\tau}{\longmapsto}_{\beta} \langle \Gamma, n \rangle \triangleright M$$

Methodology (3): Up-to Techniques

Confluence Properties

$$
\begin{array}{ccc}
\langle \Gamma, n \rangle \triangleright N & \overset{\tau}{\longmapsto}_{\beta} & \langle \Gamma, n \rangle \triangleright M \\
\mu \big\downarrow & & \\
\langle \Gamma', n' \rangle \triangleright N' & &
\end{array}
$$

Methodology (3): Up-to Techniques

Confluence Properties

$$
\begin{array}{ccc}
\langle \Gamma, n \rangle \triangleright N & \xmapsto[\beta]{\tau} & \langle \Gamma, n \rangle \triangleright M \\
\mu \downarrow & & \mu \downarrow \\
\langle \Gamma', n' \rangle \triangleright N' & \xmapsto[\beta]{\tau} & \langle \Gamma', n' \rangle \triangleright M'
\end{array}
$$

Methodology (3): Up-to Techniques

Structural Equivalence Properties

$$\langle \Gamma, n \rangle \triangleright N \quad \equiv \quad \langle \Gamma, n \rangle \triangleright M$$

Methodology (3): Up-to Techniques

Structural Equivalence Properties

$$\begin{array}{ccc} \langle \Gamma, n \rangle \triangleright N & \equiv & \langle \Gamma, n \rangle \triangleright M \\ \mu \downarrow & & \\ \langle \Gamma', n' \rangle \triangleright N' & & \end{array}$$

Methodology (3): Up-to Techniques

Structural Equivalence Properties

$$\begin{array}{ccc}
\langle \Gamma, n \rangle \triangleright N & \equiv & \langle \Gamma, n \rangle \triangleright M \\
\mu \downarrow & & \mu \downarrow \\
\langle \Gamma', n' \rangle \triangleright N' & \equiv & \langle \Gamma', n' \rangle \triangleright M'
\end{array}$$

Methodology (3): Up-to Techniques

Confluence in a Failure-free Setting

$$\Gamma, 0 \triangleright (\nu a)\,(l[\![\bar{a}]\!] \mid k[\![a.P]\!])$$

Methodology (3): Up-to Techniques

Confluence in a Failure-free Setting

$$\Gamma, 0 \triangleright (\nu a)\,(l[\![\bar{a}]\!] \mid k[\![a.P]\!]) \quad \overset{\tau}{\longmapsto}_{\beta} \quad \Gamma, 0 \triangleright (\nu a)\,(k[\![P]\!])$$

Methodology (3): Up-to Techniques

Confluence in a Failure-free Setting

$$\Gamma, 0 \triangleright (\nu a)\,(l[\![\bar{a}]\!] \mid k[\![a.P]\!]) \quad \xrightarrow{\tau}_{\beta} \quad \Gamma, 0 \triangleright (\nu a)\,(k[\![P]\!])$$

Confluence in a Failure Setting

$$\Gamma, n \triangleright (\nu a)\,(l[\![\bar{a}]\!] \mid k[\![a.P + \mathsf{susp}\ l.P]\!])$$

Methodology (3): Up-to Techniques

Confluence in a Failure-free Setting

$$\Gamma, 0 \triangleright (\nu a)\,(l[\![\bar{a}]\!] \mid k[\![a.P]\!]) \quad \xrightarrow{\tau}\!\!{}_{\beta} \quad \Gamma, 0 \triangleright (\nu a)\,(k[\![P]\!])$$

Confluence in a Failure Setting

$$\Gamma, n \triangleright (\nu a)\,(l[\![\bar{a}]\!] \mid k[\![a.P + \text{susp}\ l.P]\!]) \quad \xrightarrow{\tau}\!\!{}_{\beta} \quad \Gamma, n \triangleright (\nu a)\,(k[\![P]\!])$$

Methodology (3): Up-to Techniques

Structural Equivalence in a Failure-free Setting

$$\Gamma, 0 \triangleright l[\![P + \text{susp}\ k.Q]\!] \equiv \Gamma, 0 \triangleright l[\![P]\!] \qquad \Gamma \vdash k : \textbf{alive}$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

$$\Gamma, n \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) (l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l.R]\!])$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

$$\Gamma, n \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) (\underbrace{l_j[\![\overline{true_{i,j}}]\!]}_{\text{coordinator}} \mid \overbrace{\underbrace{l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l_j.R]\!]}_{\text{participant}}}^{\text{at round } j})$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

3 Cases

$$\Gamma, n \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) (l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l_j.R]\!])$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

Failure Free

$$\Gamma, 0 \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) \left(l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l_j.R]\!]\right)$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

Failure Free

$$\Gamma, 0 \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) (l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l_j.R]\!])$$

$$\equiv \quad (\nu true_{i,j},\ false_{i,j})\,(l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q]\!])$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

Failure Free

$$\Gamma, 0 \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) (l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l_j.R]\!])$$

$$\equiv \quad (\nu true_{i,j},\ false_{i,j})\,(l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q]\!])$$

$$\equiv \quad (\nu true_{i,j},\ false_{i,j})\,(l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P]\!])$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

Failure Free

$$\Gamma, 0 \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) (l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l_j.R]\!])$$

$$\equiv \quad (\nu true_{i,j},\ false_{i,j})\,(l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q]\!])$$

$$\equiv \quad (\nu true_{i,j},\ false_{i,j})\,(l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P]\!])$$

$$\xrightarrow{\tau}_{\beta} \quad (\nu true_{i,j},\ false_{i,j})\,(l_i[\![P]\!])$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

Broadcast same as estimate

$$\Gamma, n \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) (l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l_j.P]\!])$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

Broadcast same as estimate

$$\Gamma, n \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) (l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l_j.P]\!])$$

$$\equiv \quad (\nu true_{i,j},\ false_{i,j})\,(l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + \text{susp } l_j.P]\!])$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

Broadcast same as estimate

$$\Gamma, n \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) (l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l_j.P]\!])$$

$$\equiv \quad (\nu true_{i,j},\ false_{i,j})\,(l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + \text{susp } l_j.P]\!])$$

$$\xmapsto{\tau}_{\beta} \quad (\nu true_{i,j},\ false_{i,j})\,(l_i[\![P]\!])$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

Broadcast different from estimate

$$\Gamma, n \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) (l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l_j.R]\!])$$

Methodology (3): Up-to Techniques

Confluence Within Consensus

Broadcast different from estimate

$$\Gamma, n \triangleright \left(\nu \begin{array}{l} true_{i,j}, \\ false_{i,j} \end{array}\right) (l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + false_{i,j}.Q + \text{susp } l_j.R]\!])$$
$$\equiv \quad (\nu true_{i,j},\ false_{i,j})\,(l_j[\![\overline{true_{i,j}}]\!] \mid l_i[\![true_{i,j}.P + \text{susp } l_j.Q]\!])$$

Methodology (3): Up-to Techniques

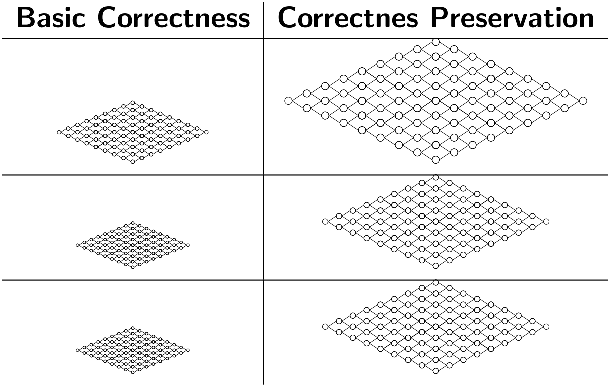
Why is it worthwhile...?

- techniques for attaining fault tolerance are bounded and reused in many algorithms.
- Fault tolerance is attained through replication!

  Space replication: $P|P|\ldots|P$

  Time replication: $P.P\ldots P$

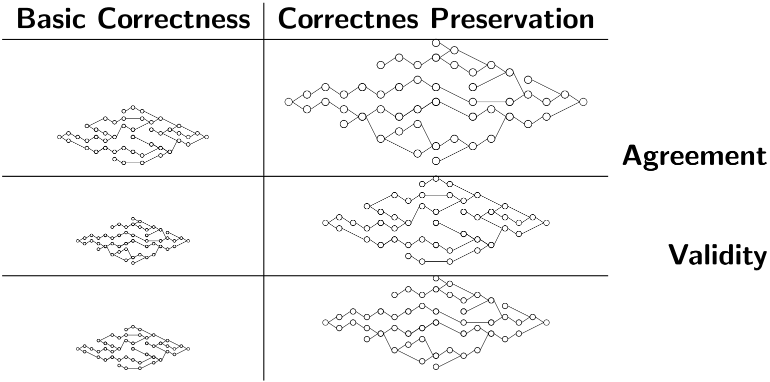
Methodology (3): ... on the bisimulation front



Agreement

Validity

Methodology (3): ... on the bisimulation front

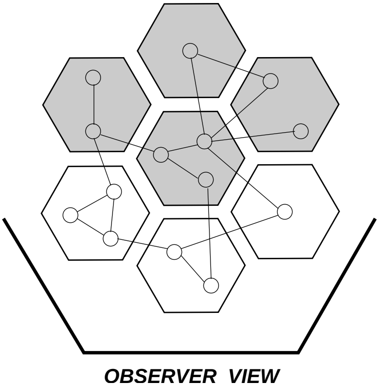


Outline

Methodology in 3 acts

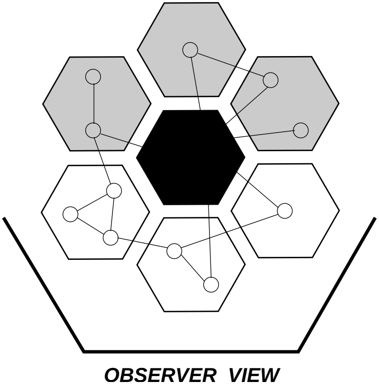
1. **Testing Harnesses:** Limiting observations to non-failing locations.
2. **Fault Tolerance:** Splitting analysis into basic correctness and correctness preservation phases.
3. **Refined Up-to Techniques:** for both failure-free and failure phases.

Fault Tolerance (...in a nut shell)



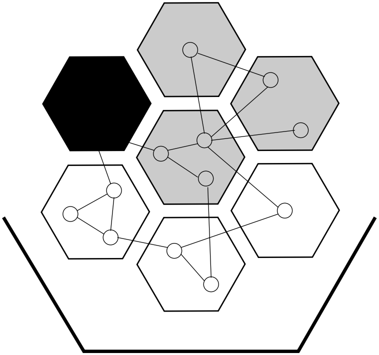
Restrict observation to immortal locations

Fault Tolerance (...in a nut shell)



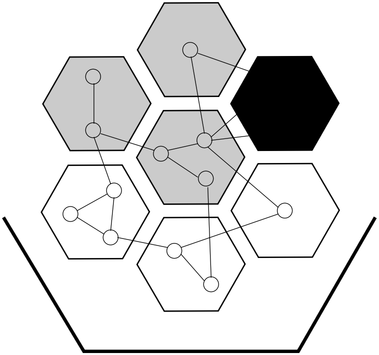
Preserves observation up to 1 failure

Fault Tolerance (...in a nut shell)



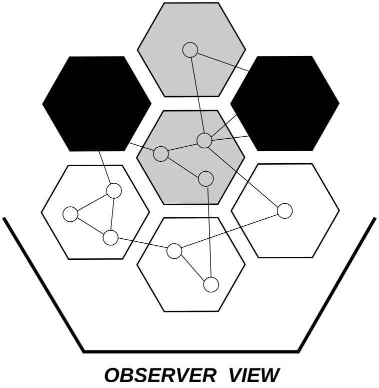
Preserves observation up to 1 failure

Fault Tolerance (...in a nut shell)



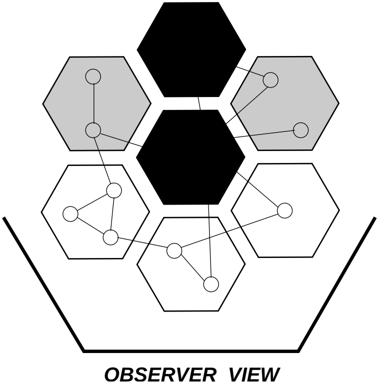
Preserves observation up to 1 failure

Fault Tolerance (...in a nut shell)



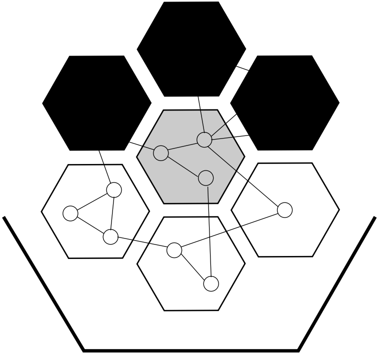
Preserves observation up to 2 failures

Fault Tolerance (...in a nut shell)



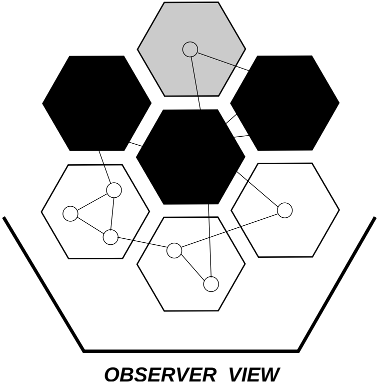
Preserves observation up to 2 failures

Fault Tolerance (...in a nut shell)



Preserves observation up to 3 failures

Fault Tolerance (...in a nut shell)



Preserves observation up to 3 failures