Complexity Bounds for Software Component Reconfiguration

Jacopo Mauro & Gianluigi Zavattaro

Rome, Sept. 2014
Mastering the Complexity of the Cloud
Models, languages and tools for the administrator of cloud applications
• Brief summary of AEOLUS Model
• What we knew so far about Deployment?
• What about Reconfiguration?
• Conclusions
Aeolus components
Bindings

Binding between two packages
Capacity constraints: upper and lower bounds to provide and require ports (resp.)
Conflicts

Conflict (no other components can activate that port)
Multi state changes

In some specific case, actions must be executed simultaneously
Actions

- Create
- Delete
- State Change
- Bind
- Unbind
Deployment problem

- **Input:**
  - A set of components to use
  - Target: component + state

- **Output:**
  - *yes* exists a plan from *empty configuration* to a configuration containing a *target*
  - *no* otherwise
Reconfiguration problem: like deployment problem but with a given initial configuration
Structure of the talk

- Brief summary of AEOLUS Model
- What we knew so far about Deployment?
- What about Reconfiguration?
- Conclusions
The deployment problem is **undecidable** in Aeolus.
Deployment problem **Polytime** without capacity constraints, conflicts, multi state change.

**Algorithm 1** Checking achievability in the Aeolus⁻ model

```plaintext
function ACHIEVABILITY(universe of resources U, resource type $\mathcal{T}$, state $q$)
    absConf := \{($\mathcal{T}', \mathcal{T}'.init) | \mathcal{T}' \in U\}
    provPort := $\bigcup_{(\mathcal{T}', q') \in absConf} \{\text{dom}(\mathcal{T}'.P_{\text{map}}(q'))\}$
    repeat
        new := \{($\mathcal{T}', q') \mid (\mathcal{T}', q'') \in absConf, (q'', q') \in \mathcal{T}'.\text{trans}\} \setminus absConf
        newPort := $\bigoplus_{(\mathcal{T}', q') \in new} \{\text{dom}(\mathcal{T}'.P_{\text{map}}(q'))\}$
        while $\exists (\mathcal{T}', q') \in new \text{ s.t. } \text{dom}(\mathcal{T}'.R_{w_{\text{map}}}(q')) \not\subseteq provPort \cup newPort$ do
            new := new \{($\mathcal{T}', q')\}
            newPort := newPort \bigcup \{\text{dom}(\mathcal{T}'.P_{\text{map}}(q'))\}
        end while
        absConf := absConf \cup new
        provPort := provPort \cup newPort
    until new = $\emptyset$
    if ($\mathcal{T}, q) \in absConf$ and $\text{dom}(\mathcal{T}.R_{w_{\text{map}}}(q')) \subseteq provPort$ then return true
    else return false
end if
end function
```
Structure of the talk

• Brief summary of AEOLUS Model
• What we knew so far about Deployment?
• **What about Reconfiguration?**
• Conclusions
Aeolus Core

- Conflicts but no capacity constraints (~ packages)
- Reconfiguration problem $\rightarrow$ Decidable
- Proof $\rightarrow$ WSTS
• $C_1 \preceq C_2$: 
  - No component $r$ in state $s$ in $C_1 \rightarrow$ No component $r$ in state $s$ in $C_2$ 
  - 1 component $r$ in state $s$ in $C_1 \rightarrow$ 1 component $r$ in state $s$ in $C_2$ 
  - $x$ components $r$ in state $s$ in $C_1 \rightarrow$ $y > x$ components $r$ in state $s$ in $C_2$
Aeolus Core

- Reconfiguration problem → **Ackerman hard**
- Proof → encoding the **Coverability** of Petri Nets with Reset Arcs
Aeolus-

- No capacity, conflicts, multi state change
- Reconfiguration problem → \textbf{PSpace}
- Proof → abstract from bindings & newly created components
• Compute possible abstract plan using non deterministic Turing machine

• Space used → polynomial

**Algorithm 1** Check reconfiguration for $C_0 = \langle U, Z_0, S, B \rangle$ and target $T_t, q_t$

```
for all $\langle T, q \rangle$ pairs in the universe $U$ do
    $B_i(\langle T, q \rangle) = C^\#_{\langle T, q \rangle}(Z_0)$
    $B_c(\langle T, q \rangle) = False$

counter $= 0$
while counter $\leq |Z_0|^k * 2^k$ do
    guess $B'_i, B'_c$
    if $\langle B_i, B_c \rangle \not\equiv \langle B'_i, B'_c \rangle$ then return False
    if $B'_i(T_t, q_t) > 0$ or $B'_c(T_t, q_t)$ then return True
    counter $= counter + 1; B_i = B'_i; B_c = B'_c$
return False
```
• Reconfiguration $\rightarrow$ PSpace hard
• Proof $\rightarrow$ encoding the reachability problem of 1-Safe Petri Nets
Structure of the talk

- Brief summary of AEOLUS Model
- What we knew so far about Deployment?
- What about Reconfiguration?
- Conclusions
Conclusions

- Deployment problem in Aeolus - polynomial → **METIS**
- Extensions of METIS for reconfiguration → yes but possibly lose completeness
- Hint to deal with reconfiguration: abstract plan with ad-hoc **heuristics**
- Other “usable” Aeolus model fragments