<u>Towards</u> an alternative characterisation of Boucherie product form

Nigel Thomas

University of Durham

Contents

- Existing methods: Boucherie
- Motivation
- Mechanisms
- Characterisations
- Conditions for simple product form
- Examples
- Conclusions

Existing methods for stochastic process algebra

- Product form
  - Reversibility / Quasi-reversibility (Harrison / Hillston)
  - Boucherie (Hillston)
  - Routing process (Sereno)
  - Queueing discipline (Clark)
- Almost product form
  - Time scale decomposition (Mertziotakis / Hillston)
  - Synchronisation points (Haverkort / Bohenkamp)
  - Decision free processes (Mertziotakis)
  - Near independence (Ciardo / Trivedi)
  - …

Boucherie Product Form

- Based on a number of components which only interact over a resource:
  - Certain actions can only happen with the resource.
  - If one component is using the resource another cannot, <u>but may do something else</u>.
  - The resource is explicit and redundant.

Motivation

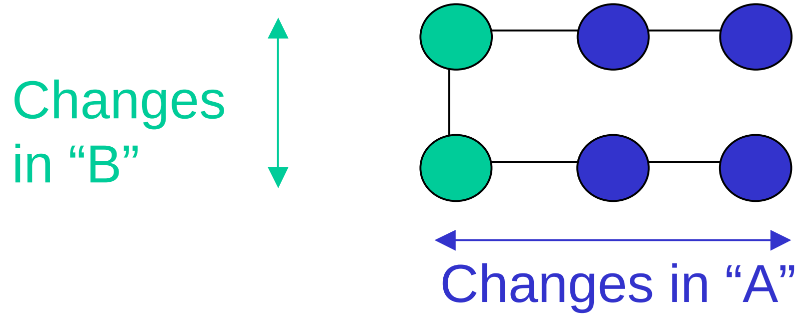
- Investigate intersection between different criteria for product form.
- Allow characterisation of Boucherie type models <u>without</u> an explicit resource.
- Investigate whether PEPA is really sufficient for this task.

Mechanisms

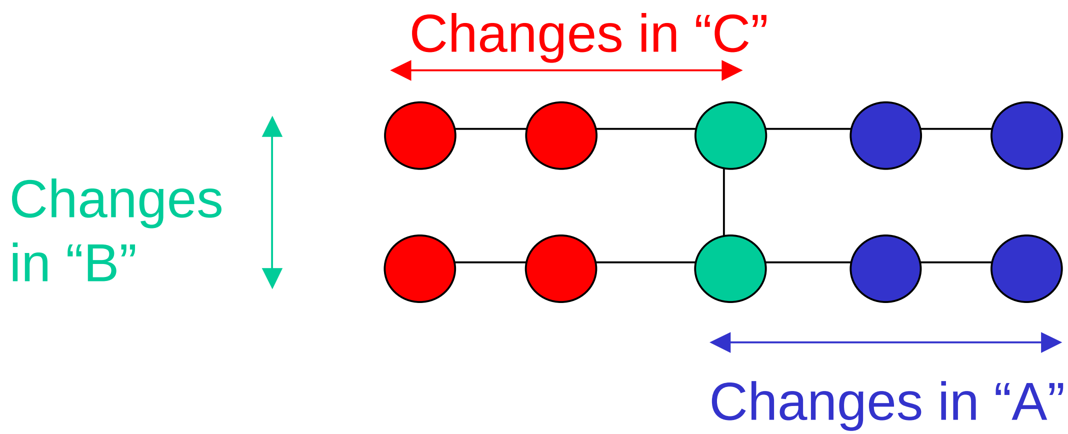
- Behavioural Independence.
- Partial Behavioural Independence.
- Restricted Partial Behavioural Independence.

Characterisations

- The simplest characterisation identifies models with the following structure:

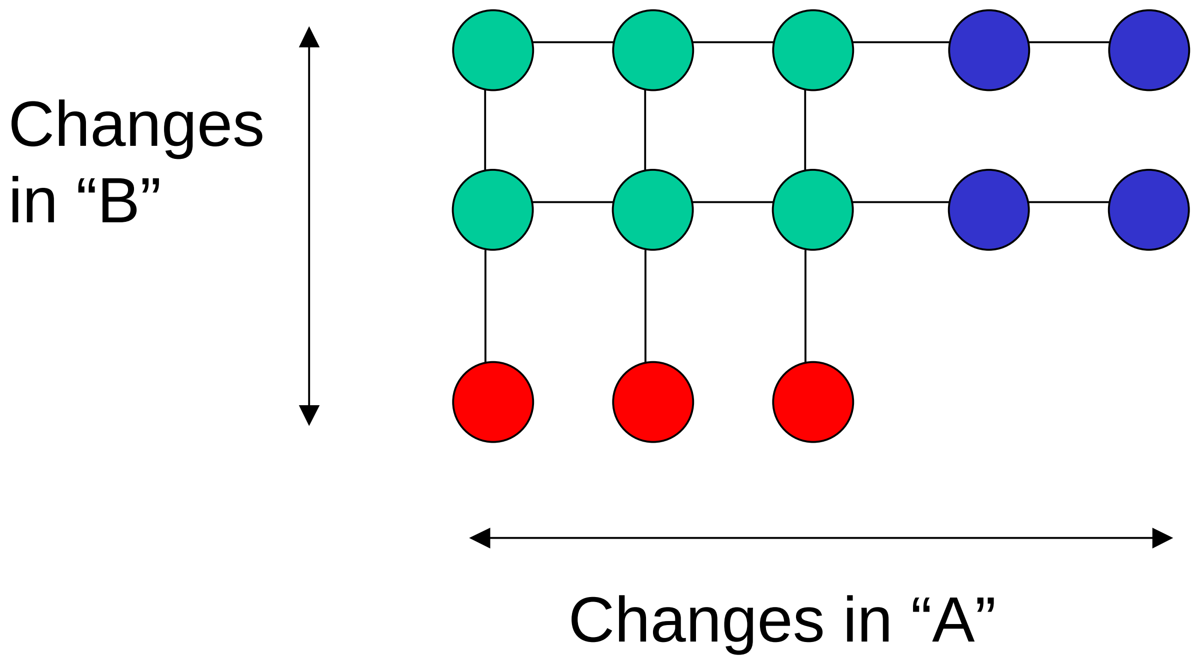


- Partial behavioural independence allows characterisation over multiple components:



Characterisations (2)

• Restricted partial behavioural independence allows characterisation over multiple interacting components:



Conditions for simple product form

1. One component, A, of a pair $A \underset{L}{\bowtie} B$ is behaviourally independent.
2. The other component, B, is controlled by A over all the actions in the cooperation set, $\mathcal{K}(B) = L$.
3. The complete action type set of B, $\vec{\mathcal{A}}(B)$ is contained within its interface, $\vec{\mathcal{A}}(B) = L$.
4. All actions in the cooperation set, L, are enabled in exactly one derivative of A.
5. No actions in the cooperation set, L, are enabled in any other derivative of A.
6. $\vec{\mathcal{A}}_f(A) \cap \vec{\mathcal{A}}_f(B) = \emptyset$

Product form over components

$$
\begin{aligned}
Queue_0 &\stackrel{def}{=} (arrival, \top).Queue_1 \\
Queue_i &\stackrel{def}{=} (arrival, \top).Queue_{i+1} + (service, \top).Queue_{i-1} \\
& \quad 1 \leq j \leq N-1 \\
Queue_N &\stackrel{def}{=} (service, \top).Queue_{N-1} \\
\\
Server_{on} &\stackrel{def}{=} (fail, \xi).Server_{off} + (arrival, \lambda).Server_{on} + (service, \mu).Server_{on} \\
Server_{off} &\stackrel{def}{=} (repair, \eta).Server_{on}
\end{aligned}
$$

$$
Queue_0 \underset{\{service, arrival\}}{\bowtie} Server_{on}
$$

$$
\begin{aligned}
Queue_0 &\stackrel{def}{=} (arrival, \top).Queue_1 \\
Queue_i &\stackrel{def}{=} (arrival, \top).Queue_{i+1} + (service, \top).Queue_0 \\
&\quad 1 \leq j \leq N-1 \\
Queue_N &\stackrel{def}{=} (service, \top).Queue_0
\end{aligned}
$$

$$
\begin{aligned}
Server_{on} &\stackrel{def}{=} (fail, \xi).Server_{off} + (arrival, \lambda).Server_{on} \\
&\quad +(service, \mu).Server_{on} \\
Server_{off} &\stackrel{def}{=} (repair1, \eta_1).Server_{standby} \\
Server_{standby} &\stackrel{def}{=} (repair2, \eta_2).Server_{on}
\end{aligned}
$$

$$
Queue_0 \underset{\{service, arrival\}}{\bowtie} Server_{on}
$$

Security Guards Example (1)

- 2 guards, one must be on duty at all times, the other may sleep.

$$G_A Awake \stackrel{def}{=} (aFallAsleep, r_1).G_A Asleep + (bFallAsleep, \top).G_A Awake$$

$$G_A Asleep \stackrel{def}{=} (wakeup, r_2).G_A Awake$$

$$G_B Awake \stackrel{def}{=} (bFallAsleep, r_3).G_B Asleep + (aFallAsleep, \top).G_B Awake$$

$$G_B Asleep \stackrel{def}{=} (wakeup, r_4).G_B Awake$$

$$G_A Awake \underset{\{aFallAsleep, bFallAsleep\}}{\bowtie} G_B Awake$$

Security Guards Example (2)

- Guards may patrol together.

$$G_A Awake \stackrel{def}{=} (goOut, r_5).G_A Patrol + (aFallAsleep, r_1).G_A Asleep$$
$$+ (bFallAsleep, \top).G_A Awake$$

$$G_A Patrol \stackrel{def}{=} (goBack, r_6).G_A Awake$$

$$G_B Awake \stackrel{def}{=} (goOut, r_7).G_B Patrol + (bFallAsleep, r_3).G_B Asleep$$
$$+ (aFallAsleep, \top).G_B Awake$$

$$G_B Patrol \stackrel{def}{=} (goBack, r_8).G_B Awake$$

$$G_A Awake \underset{\substack{\{goOut, goBack, \\ aFallAsleep, bFallAsleep\}}}{\bowtie} G_B Awake$$

Security Guards Example (3)

- One Guard patrols but the other must be awake (r_5 must equal r_7).

$$G_A Awake \stackrel{def}{=} (goOut, r_5).G_A Patrol + (aFallAsleep, r_1).G_A Asleep$$
$$+ (goOut, \top).G_A Awake + (bFallAsleep, \top).G_A Awake$$

$$G_A Patrol \stackrel{def}{=} (goBack, r_6).G_A Awake$$

$$G_B Awake \stackrel{def}{=} (goOut, r_7).G_B Patrol + (bFallAsleep, r_3).G_B Asleep$$
$$+ (goOut, \top).G_B Awake + (aFallAsleep, \top).G_B Awake$$

$$G_B Patrol \stackrel{def}{=} (goBack, r_8).G_B Awake$$

$$G_A Awake \underset{\{goOut,\ aFallAsleep,\ bFallAsleep\}}{\bowtie} G_B Awake$$

Security Guards Example (4)

- Guards may sleep regardless.

$$G_A Awake \stackrel{def}{=} (goOut, r_5).G_A Patrol + (aFallAsleep, r_1).G_A Asleep$$
$$+ (bFallAsleep, \top).G_A Awake$$

$$G_A Patrol \stackrel{def}{=} (goBack, r_6).G_A Awake + (bFallAsleep, \top).G_A Patrol$$

$$G_B Awake \stackrel{def}{=} (goOut, r_7).G_B Patrol + (bFallAsleep, r_3).G_B Asleep$$
$$+ (aFallAsleep, \top).G_B Awake$$

$$G_B Patrol \stackrel{def}{=} (goBack, r_8).G_B Awake + (aFallAsleep, \top).G_B Awake$$

$$G_A Awake \underset{\{aFallAsleep, bFallAsleep\}}{\bowtie} G_B Awake$$

Further work

- Generalisation to full Boucherie product form.
- Formal relationships between decomposition methods.
- Efficient application of methods.
- Partial evaluation, real-time solution, …
- Applications…