Nomadic Time

Andrew Hughes
http://www.dcs.shef.ac.uk/~andrew

Department of Computer Science
University of Sheffield

RelMiCS - 29/08/2006

Outline

1. Introduction

Outline

1 Introduction

2 A Simple Example

Outline

1. Introduction
2. A Simple Example
3. Further Thoughts and Conclusions

CCS

- The *Calculus of Communicating Systems* models processes and the interactions which take place between them.
- Interactions are modelled via sequences of *actions*.
- When one process performs an action, o, and another process concurrently performs the co-action, $\overline{o}$, the two may *synchronize*.
- The two actions take place simultaneously, resulting in a silent action (denoted by a τ).
- Action names are commonly used to represent *channels*.
- The two variants, $\overline{o}$ and o represent sending and receiving, respectively.

Scaling Synchronization

Example

$$o.E \mid \overline{o}.F$$

- Easy to do *local synchronization* in CCS – one sender, one receiver.
- But what about with an arbitrary number (n) of processes? (*global synchronization*)
- Can be done, but *not compositionally*

The Problem

Example

$$\overline{o}.\overline{o}.E \mid o.F \mid o.G$$

- The case with two receivers works fine. . .

The Problem

Example

$$\overline{o}.\overline{o}.\overline{o}.E \mid o.F \mid o.G \mid o.H$$

- But further composition requires rebuilding the semantics of the sender.

How Do We Fix This?

- To send multiple times, recursion is needed.
- But what is the base case of this recursion?
- When all possible synchronizations have occurred.
- How is this determined?
- Timed calculi, like the *Calculus of Synchronous Encapsulation* (CaSE), provide a solution.

The Solution

Example

$$\mu X.\lfloor \overline{o}.X \rfloor \sigma(P) \mid o.E \mid o.F \mid o.G$$

- Use of the timeout operator, $\lfloor E \rfloor \sigma(F)$ – perform F if E times out on σ.
- Recursive output with the clock signal effectively the base case.
- Clock will tick when no more synchronizations can occur.
- *Maximal progress* gives silent actions precedence over clock ticks.

Mobility

- But timed calculi can only handle *static systems*.
- What about a situation where a process may change its location during execution?
- In contrast, the *ambient calculus* provides distribution and mobility.
- But suffers the same deficiency as CCS with respect to global synchronization.

Typed Nomadic Time

- Combines CaSE with notions of distribution and mobility from the ambient calculus and its variants.
- Allows the creation of compositional semantics for mobile component-based systems.
- Broadcasts can be localised to a changing group of processes.

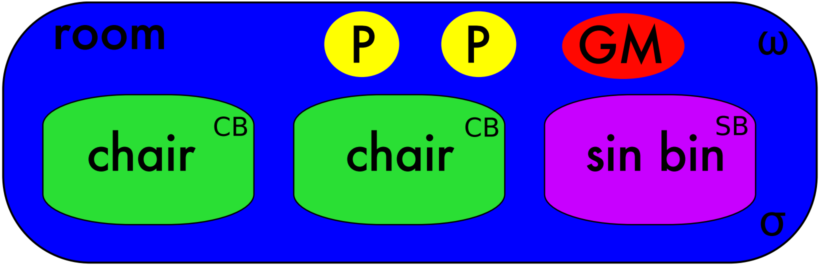
Modelling Musical Chairs

1. The players begin the game standing. The number of players is initially equal to the number of chairs.
2. The music starts.
3. A chair is removed from the game.
4. The music stops.
5. Each player attempts to obtain a chair.
6. Players that fail to obtain a chair are out of the game.
7. The music restarts. Any players who are still in the game leave their chairs and the next round begins (from stage three).

The Game Environment

- Represented using named locations (*localities*)
- These can be nested to form a forest structure.
- Each chair is a *locality*.
- The 'sin bin' is also a *locality*.
- Encapsulated in a top-level *room* locality for a cleaner solution.

The Game Environment



The Game Environment

Example

$$room[chair[\mathbf{0}]_{\emptyset}^{CB} \mid chair[\mathbf{0}]_{\emptyset}^{CB}]_{\{\sigma\}}^{\omega}.$$

- **0** is a process with no explicit behaviour.
- σ is a clock.
- *CB* and ω are *bouncers*.

Clocks

- The presence of music is signified by the ticks of a clock, σ.
- Also signifies the implicit acknowledgement that all available chairs have been taken.
- The clock appears on the bottom right to indicate that its ticks are visible within the locality, but not outside.
- Ticks become silent actions outside location boundaries.

Bouncers

- The locality manager. Named after the person who stands outside a nightclub.
- Dictates whether processes are allowed to enter or exit.
- Also controls whether the locality may be destroyed.
- For the room, protection is irrelevant, so ω allows everything.

Definition

$$\omega \stackrel{\text{def}}{=} \mu X.(\overline{in}.X + \overline{out}.X + \overline{open}.X)$$

Bouncers

- The chair bouncer, *CB*, enforces the implicit one-person-per-chair predicate.

Definition

$$CB \stackrel{\text{def}}{=} \mu X.(\overline{in}.\overline{out}.X + \overline{open})$$

Bouncers

- The sin bin bouncer, *SB*, prevents players getting back out.

Definition

$$SB \stackrel{\text{def}}{=} \mu X.\overline{in}.X$$

Compositional Movement

- Central to the use of TNT is the *compositional* movement of players to chairs.
- A *gamesmaster* process broadcasts the movement directive.
- This works regardless of the number of players and chairs involved.

Example

$$\mu X.(\lceil in\ chair\ sit.X \rceil \sigma(GM6)) \mid \lceil sit.PChair \rceil \sigma(Loser) \mid chair[\mathbf{0}]^{CB}_{\emptyset}$$

Multiway Synchronization

- For the player to actually enter the chair, the following actions must take place simultaneously:
  - The gamesmaster must perform *in chair sit*.
  - The player must synchronize with this on *sit*.
  - The chair bouncer must allow the player in, via $\overline{in}$.

Example

$$\mu X.(\lceil in\ chair\ sit.X \rceil \sigma(GM6)) \mid \lceil sit.PChair \rceil \sigma(Loser) \mid chair[\mathbf{0}]^{CB}_{\emptyset}$$

Multiway Synchronization

If this happens, a τ action occurs and:

Example

$$\mu X.(\lceil in\ chair\ sit.X \rceil \sigma(GM6)) \mid \lceil sit.PChair \rceil \sigma(Loser) \mid chair[\mathbf{0}]^{CB}_{\emptyset}$$

evolves to become:

Example

$$\mu X.(\lceil in\ chair\ sit.X \rceil \sigma(GM6)) \mid chair[\mathbf{0} \mid PChair]^{\overline{out}.CB}_{\emptyset}$$

Handling The Losers

- Losing players are moved to the sin bin in much the same way.
- The difference is in the use of *localized broadcast*.
- There is no inter-locality communication.
- This ensures that only players still in the room and not in a chair will be able to synchronize.

Conclusions

- A novel combination of features, where arbitrary numbers of agents can synchronize and move around a dynamic topology.
- An operational semantics exists for the calculus.
- Currently refining a type system, which enables further movement control.
- Future work will consider more detailed case studies (e.g. quorum sensing in bacteria) and possible stochastic extensions.

The End

Thanks for listening.
Any questions?

CARDELLI, L., GORDON, A.D.
Mobile Ambients.
In *Proceedings of the 1st International Conference on Foundations of Software Science and Computation Structures (FoSSaCS '98)* (1998), no. 1378 in LNCS, Springer-Verlag.

MILNER, R.
Communication and Concurrency
Prentice-Hall, London (1989)

NORTON, B., LÜTTGEN, G., AND MENDLER, M.
A Compositional Semantic Theory for Synchronous Component-based Design.
In *Proceedings of the 14th International Conference on Concurreny Theory (CONCUR '03)* (2003), no. 2761 in LNCS, Springer-Verlag.