Implementing CRSM Rendezvous Communication for a Distributed Robotic System

B. Election Reddy
07305054

under the guidance of
Prof. Kavi Arya and Prof. Ramesh S

June 23, 2009
Outline

1. Concurrent Languages
   - Synchronous Languages
   - Asynchronous Languages
   - CRSM

2. Fire Bird Platform

3. CRP Rendezvous

4. Rendezvous Implementation

5. Network Problems

6. Case Study: Treasure Hunt

7. Summary

8. Future Work
Concurrent Languages

Synchronous Languages

- Deterministic
- Logical Concurrency
- Suits for describing Centralized Controllers
- eg. Esterel, Argos
Asynchronous Languages

- Non-Deterministic
- Physical Concurrency
- Suits for describing Distributed Controllers
- eg. CSP, ADA
How to describe GALS???
Concurrent Languages

How to describe GALS???

Communicating Reactive State Machines (CRSM)
CRSM

- Language for describing behaviors of distributed controllers
- Individual nodes are described using Argos
- Inter node communication is done using Rendezvous Communication primitives
- Constructs
  - Hierarchical Composition
  - Synchronous Parallel Composition
  - Signal Hiding
  - Asynchronous Composition
Figure: Hierarchical Composition
Figure: Synchronous Parallel Composition
Implementing CRSM Rendezvous Communication for a Distributed Robotic System

**Figure**: Asynchronous Composition
Fire Bird Platform

- Platform: Firebird running uC/OS. uC/OS is an Operating System Kernel for microcontrollers
  - Real Time
  - Multitasking
  - Preemptive Priority Based Scheduling
- Communication: XBee wireless

Figure: Firebird 4
CRP Rendezvous[2]

Solution is non-trivial due to

- **Facts:**
  - **A1:** Communication takes non-zero time
  - **A2:** Reactive processes are invoked at arbitrary points of times

- **Requirements:**
  - **P1 Safety:** Coordinators, commit to only those rendezvous that are enabled. Further, they agree on the commitment of rendezvous.
  - **P2 Liveness:** It will not be the case that no rendezvous is committed when there is a rendezvous that is enabled for ever.
There is no protocol with the assumptions $A_1$ and $A_2$ satisfying the properties $P_1$ and $P_2$ [3].
There is no protocol with the assumptions \( A_1 \) and \( A_2 \) satisfying the properties \( P_1 \) and \( P_2 \) \[3\]

*Solution by CRP*: By weakening the assumption \( A_2 \) by giving control to the reactive process only at definite points of time
CRP Rendezvous[2] [cont’d…]

- Reactive Process
- Coordinator Process
- Signals
  - \(\text{Start}(S_r)\): request signal output by the node (reactive process) when it is ready to execute rendezvous \(r\)
  - \(\text{Return}(R_r)\): input signal generated by the coordinator to indicate to the node that it can go ahead with the completion of the execution of the rendezvous \(r\)
  - \(\text{Kill}(K_r)\): signal output by the node when a local reaction preempts the rendezvous \(r\)
Commitment Sequence

- Consent phase
- Confirmation phase

Figure: Coordinator State Diagram[2]
Status of a rendezvous $r$

- **closed**: the reactive node is not ready to execute $r$
- **untried**: the reactive node is ready to execute $r$ but the coordinator is yet to execute the commitment sequence for $r$
- **trying**: the coordinator is in the middle of executing the commitment sequence for $r$
- **rejected**: the execution of commitment sequence for $r$ is failed
- **committed**: the commitment sequence for $r$ is executed successfully
Rendezvous Implementation

API

- void rendezvousSend(INT8U from, INT8U to, INT8U channel)
- INT8U rendezvousReceive(INT8U from, INT8U to, INT8U channel)
Rendezvous Implementation

Assumptions about the network

- The nodes are connected to each other via point-to-point links that do not fail.
- The links are bidirectional and the coordinators use these to communicate with each other.
- The network is reliable i.e., messages are not lost and their order is preserved during transmission.
- The nodes are numbered with distinct indices and each node knows its index.
Implementing CRSM Rendezvous Communication for a Distributed Robotic System

CoordinatorTask

```
CoordinatorTask
```

```
Concurrent Languages
Fire Bird Platform
CRP Rendezvous
Rendezvous Implementation
Network Problems
Case Study: Treasure Hunt
Summary
Future Work
```
Rendezvous Implementation

**Figure**: CRP Rendezvous Implementation using uC/OS

B. Election Reddy 07305054

Implementing CRSM Rendezvous Communication for a Distributed Robotic System
Network Problems

- Message Losses and Reordering
  ZigBee 802.15.4 protocol guarantees the reliable communication. So we can safely assume that there won’t be any message losses and reordering.

- Interference
  - CSMA/CA option of XBee module is enabled. So there is very less chances of interference
CSMA/CA

- Sender node senses the medium before sending the data
- If the medium is clear then sends the data
- Otherwise waits for a random amount of time (backoff period) and then checks the medium again
- If the medium is still busy after backoff period then chooses other backoff period and repeats the same
What happens if more than one node senses the medium at the same time???
What happens if more than one node senses the medium at the same time???

Exception Handling
Corrupted messages are handled as below

- If a Task receives a corrupted message while protocol is running, it just discards that message, sends *IGNORE* message to the partner and restarts the rendezvous.
- If a Task receives a *IGNORE* message it discards that message and restarts the rendezvous.
- So, the state of the robot will be same before and after a corrupted message is received.
Case Study: Treasure Hunt

- F1, F2, F3 are three Fire Birds
- All three will go in their own direction for finding treasure
- If any one of them finds the treasure, it will stop there and sends the *TreasureFound* message to others
- *TreasureFound* message receptor will stop moving in its own direction and goes to the *TreasureFound* message sender
- *Assumption*: Only one treasure exist in the system
Implementing CRSM Rendezvous Communication for a Distributed Robotic System

Node1

Figure: CRSM Representation of Node1

B. Election Reddy 07305054

Implementing CRSM Rendezvous Communication for a Distributed Robotic System
Concurrent Languages
Fire Bird Platform
CRP Rendezvous
Rendezvous Implementation
Network Problems
Case Study: Treasure Hunt
Summary
Future Work

Node2

Node N2

Task1
- noTreasure/
- treasure/stop
- found/
  - MoveForward
  - CH2 !
  - CH4 !
  - Stand

Task2
- CH1 ?
- /found
  - stop/
  - GoTo N1

Task3
- CH6 ?
- /found
  - stop/
  - GoTo N3

Implementing CRSM Rendezvous Communication for a Distributed Robotic System

B. Election Reddy 07305054
Node3

Node N3

Task 1
- noTreasure/
  - MoveForward
  - treasure/stop
    - CH5
      - CH6
        - Stand

Task 2
- CH3 ?
  - /found
    - stop/
    - GoTo N1

Task 3
- CH4 ?
  - /found
    - stop/
    - GoTo N2

Implementing CRSM Rendezvous Communication for a Distributed Robotic System
Summary

- Understood the CRSM concepts and using of the CRSM tools
- Acquainted with the programming environment of Robotic System (Fire Bird IV) running Real-Time kernel uCOS
- Implemented the CRP Rendezvous Communication primitives for Fire Bird using uC/OS
Future Work

- CRP Rendezvous communication protocol is taking approximately 500 milliseconds. This can be improved by tuning the delays involved in its implementation.
- Implementing the CRSM to C translator using this communication primitives
- Integrating CRP rendezvous implementation with Esterel2C implementation by Shashidhar[15]
Conclusion

We have implemented CRP Rendezvous Communication primitives for a distributed robotic system so that the robots can communicate synchronously.
Thank you.
References

www.nex-robotics.com

Implementation of Communicating Reactive Processes by S. Ramesh, Parallel Computing 25(6), 1999


References

- *Model Based Languages for Distributed Systems*, DD Dissertation by Kumar Digvijay Singh, 2007
References

References

