Routing Protocols in Mobile Ad-hoc Networks

Michael Westergaard
mw@daimi.au.dk

Department of Computer Science
University of Aarhus
Overview

- A project on routing in mobile ad-hoc networks
- Modules in coloured Petri-nets
- Demo
The Project

- Participants: Ericsson Telebit, CPN Group at AU
- Project duration: July 2003 - December 2005
- Executive summary summary: This project deals with the design and validation of routing protocols and other protocols in ad-hoc and mobile networks
- Project home-page: http://www.daimi.au.dk/CPnets/IPv6/
Mobile Ad-hoc Networks

AHN(3) -> 3ffe:100:3:405::3
AHN(4) -> 3ffe:100:3:405::4
AHN(5) -> 3ffe:100:3:406::5

DNS Server
3ffe:100:3:401::1

Gateway 1
3ffe:100:3:401::3
3ffe:100:3:405::1

Gateway 2
3ffe:100:3:401::4
3ffe:100:3:406::1

DNS request/reply/update
Gateway advertisement
Data packet

Ad-hoc Node 3
3ffe:100:3:405::3

Ad-hoc Node 4
3ffe:100:3:405::4

Ad-hoc Node 5
3ffe:100:3:406::5

Host 1
3ffe:100:3:401::2
Mobile Ad-hoc Networks

We can move
Mobile Ad-hoc Networks

We can move
- ad-hoc nodes
Mobile Ad-hoc Networks

We can move ad-hoc nodes
Mobile Ad-hoc Networks

We can move ad-hoc nodes
Mobile Ad-hoc Networks

We can move
- ad-hoc nodes

- AHN(3) -> 3ffe:100:3:405::3
- AHN(4) -> 3ffe:100:3:405::4
- AHN(5) -> 3ffe:100:3:406::5

- DNS request/reply/update
- Gateway advertisement
- Data packet
Mobile Ad-hoc Networks

We can move
- ad-hoc nodes
We can transmit data
Mobile Ad-hoc Networks

- We can move ad-hoc nodes
- We can transmit data along the lines
Mobile Ad-hoc Networks

We can move
- ad-hoc nodes

We can transmit data
- along the lines
- through the cloud
Mobile Ad-hoc Networks

We can move
- ad-hoc nodes

We can transmit data
- along the lines
- through the cloud
- but not too far
Real-life Application

- Modern mobile phone (GPRS or 3G) connected to a service provider, e.g. streaming music
- Sitting in a moving train
- Want the music to play continuously – even when moving from one antenna to another
- 54 places
- 40 transitions
- 54 places
- 40 transitions
- = big
CPN Model

- 54 places
- 40 transitions
- = big
- = incomprehensible?
- 54 places
- 40 transitions
- = big
- = incomprehensible?
- No because we use modules
- 54 places
- 40 transitions
- = big
- = incomprehensible?
- No because we use modules
- 19 different modules
```java
int radius = 5;
int exponent = 2;
int radiusSquare = 1;
for (int i = 0; i < exponent; i++) {
    radiusSquare = radiusSquare * radius;
}
double area = radiusSquare * 3.1415926
```
```java
int radius = 5;
int exponent = 2;
int radiusSquare = 1;
for (int i = 0; i < exponent; i++) {
    radiusSquare = radiusSquare * radius;
}
double area = radiusSquare * 3.1415926
```
```java
int radius = 5;
int exponent = 2;
int radiusSquare = 1;
for (int i = 0; i < exponent; i++) {
    radiusSquare = radiusSquare * radius;
}
double area = radiusSquare * 3.1415926
```
int power(int value, int exponent) {
    int result = 1;
    for (int i = 0; i < exponent; i++) {
        result = result * value;
    }
    return result;
}

int radius = 5

double area = power(radius, 2) * 3.141592
A Coloured Petri-net

1`1(1,"Modelling")++
1`2(2,"g and An")++
1`3(3,"alysis b")++
1`4(4,"y Means ")++
1`5(5,"of CPN ")++
1`6(6,"red Petr")++
1`7(7,"i Nets##")
A Coloured Petri-net
A Simpler Coloured Petri-net

1. (1, "Modelling")++
2. (2, "g and An")++
3. (3, "alysis b")++
4. (4, "y Means ")++
5. (5, "of CPN ")++
6. (6, "red Petr")++
7. (7, "i Nets##")
Main Module

if success then 1\`(n,d) else empty

Receiver

Sender

Transmit Packet

D

INT

Transmit Ack

C

INT

n

A

INT\times DATA

(n,d)

B

INT\times DATA

if success then 1\`n else empty

Sender

Receiver
Advantages of Modules

- We can split up our program/model into smaller, more comprehensible parts
- We can re-use sub-modules
  \[ c = \text{squareRoot}(\text{power}(a, 2) + \text{power}(b, 2)) \];
- We can replace submodules

```java
int power(int v, int e) {
    return (int) Math.pow(v, e);
}
```
Host 1 wants to send data to Ad-hoc Node 3
Demo
Conclusions

- Coloured Petri-nets can cope with large, realistic models if we use modules
- The industry is interested in models
  - Easier to control and reproduce scenarios
  - Implementation details can be abstracted away