

# Κατανεμημένα Συστήματα I

## Μάθημα Βασικής Επιλογής,

### Χειμερινού Εξαμήνου

#### Τομέας Εφαρμογών και Θεμελιώσεων

Χρήστος Κονίνης

Τρίτη, 10 Νοεμβρίου, 2009  
Υπολογιστικό



## Επισκόπηση

### Controlling the simulation

Collecting Messages Statistics

Collecting Messages Statistics Examples

### Configuration of an Application

Passing parameters in the processor

Passing parameters Example

JShawn

JShawn

3η Άσκηση

FloodMax



## transmission model

- ▶ Μέχρι τώρα κατασκευάζαμε ένα simulation world με την εντολή:
  - ▶ `prepare_world edge_model=simple comm_model=disk_graph range=1`
- ▶ Στην οποία αν και δεν δηλώνεται, το transmission model είναι το reliable, δηλαδή όλα τα μηνύματα αποστέλλονται με επιτυχία και χωρίς καθυστερήσεις
  - ▶ `prepare_world edge_model=simple comm_model=disk_graph transm_model=reliable range=1`
- ▶ Γενικά μπορούμε να χρησιμοποιήσουμε πολλαπλά transmission models και κάθε μήνυμα που αποστέλλεται περνάει από κάθε transmission model πριν παραδοθεί στους τελικούς παραλήπτες
  - ▶ `Message -> ChainTransModel1 -> ChainTransModel2 -> EndTransModel`



## Collecting Messages Statistics

- ▶ Αν θέλουμε ο shawn να κρατάει στατιστικά για τον αριθμο των μηνυμάτων που στάλθηκαν σε μια εξομοίωση, χρησιμοποιούμε το stats\_chain μοντέλο

```
prepare_world edge_model=list comm_model=disk_graph \  
transm_model=stats_chain \  
range=1 \  
chain_transm_model name=reliable
```

- ▶ Προσθέσαμε μια επιπλέον παράμετρο `transm_model=stats_chain`, που συλλέγει πληροφορίες για το σύνολο των μηνυμάτων που στέλνονται
- ▶ Το `task chain_transm_model` προσθέτει το μοντέλο `reliable` που τελικά μεταδίδει τα μηνύματα
  - ▶ `Message -> stats_chain -> reliable`



## Collecting Statistics Tasks

- ▶ Για να πάρουμε τα στατιστικά που συλλέγει το stats\_chain μοντέλο χρησιμοποιούμε 2 tasks:
  - ▶ connectivity :  
Τυπώνει τον μέσο αριθμό γεγόνων των κόμβων καθώς και τον μέγιστο και ελάχιστο αριθμό γεγόνων
  - ▶ dump\_transmission\_stats :  
Τυπώνει τον συνολικό αριθμό μηνυμάτων που στάλθηκαν, καθώς και αναλυτικά τον αριθμό των διαφορετικών τύπων μηνυμάτων



## Collecting Messages Statistics Example

1. Φτιάχνουμε ένα αρχείο msgstats.conf στον φάκελο shawn/buildfiles και γράφουμε τις παρακάτω εντολές :

```
prepare_world edge_model=list comm_model=disk_graph \  
  transm_model=stats_chain \  
  range=1 \  
  chain_transm_model name=reliable \  
  
rect_world width=50 height=50 count=5000 processors=helloworld \  
simulation max_iterations=10 \  
  
connectivity \  
dump_transmission_stats
```

2. Ανοίγουμε μια κονσόλα στον φάκελο shawn/buildfiles
3. Εκτελούμε την εντολή:  
./shawn -f msgstats.conf



## Επισκόπηση

### Controlling the simulation

- Collecting Messages Statistics
- Collecting Messages Statistics Examples

### Configuration of an Application

- Passing parameters in the processor
- Passing parameters Example

### JShawn

JShawn

### 3η Άσκηση

FloodMax



## Writing parameters in the conf file

- ▶ Σε πολλές εξομοιώσεις είναι χρήσιμο να θέτουμε μεταβλητές στο .conf αρχείο και να τις διαβάζουμε μέσα από τον processor
- ▶ Έτσι μπορούμε να αλλάζουμε τις παραμέτρους του αλγορίθμου χωρίς να κάνουμε recompile το shawn
- ▶ Η γενική μορφή ανάθεσης τιμής σε μεταβλητή είναι:  
<variable\_name>=<variable\_value>
- ▶ Οι μεταβλητές μπορεί να είναι integers, strings, boolean , floats:

### Παράδειγμα

```
my_int_value=1  
my_string_value>HelloWorld  
my_boolean_value=true  
my_float_value=3.14
```



## Reading parameters in the processor

- ▶ Όταν ο shawn διαβάζει το config αρχείο, θέτει τις μεταβλητές στο Simulation Environment και μπορούν να προσπελαστούν χρησιμοποιώντας τις μεθόδους:
  - ▶ `required_string_param( 'variable_name' )`:  
Επιστρέφει την τιμή της παραμέτρου <variable\_name>, αν δεν υπάρχει προκαλεί μια εξήρεση (exception)
  - ▶ `optional_string_param( 'variable_name', DEFAULT_VALUE )`:  
Επιστρέφει την τιμή της παραμέτρου <variable\_name>, αν δεν υπάρχει επιστρέφει την τιμή DEFAULT\_VALUE

### Παράδειγμα

```
int integer_param_;  
const shawn::SimulationEnvironment& se =  
    owner().world().simulation_controller().environment();  
integer_param_ = se.optional_int_param( 'integer_param_name', 0 );
```



## Passing parameters Example

- ▶ Θα τροποποιήσετε το `fr02_Processor.cpp` στο `shawn/src/legascyapps/fr02` ώστε να στέλνει το πρώτο μήνυμα στον γύρο N που θα ορίζεται από μια παράμετρο `send_sound` στο config αρχείο.
- ▶ Αν δεν είχατε ολοκληρώσει την άσκηση μπορείτε να κατεβάσετε τον κώδικα από εδώ:

```
ftp://carrot.cti.gr/fr02/fr02_partA.tar.bz2
```



## Passing parameters Example

1. Προσθέστε την παράμετρο `send_round=3` στην αρχή του `fr02-config.conf`
2. Προσθέστε μια μεταβλητή `int send_round_` στο `fr02_Processor.h`
3. Στην μέθοδο `boot()` του `fr02_Processor.cpp` διαβάζουμε την παράμετρο `send_round` που θέσαμε στο config:

```
const shawn::SimulationEnvironment& se =  
    owner().world().simulation_controller().environment();  
send_round_ = se.optional_int_param( 'send_round', 0 );
```

4. Βρίσκουμε τα σημεία που Gateway στέλνει hello μηνύματα και προσθέτουμε ένα έλεγχο, εξετάζοντας αν ο γύρος εξομίσωσης είναι ίσος με `send_round_`

```
int simulation_round(): Returns the current simulation round
```

5. Εκτελούμε `make` για να κάνουμε `compile` το shawn
6. Εκτελούμε την εξομίσωση:  
`./shawn -f fr02-config.conf`
7. Πειραματιστείτε με την τιμή της `send_round` στο `fr02-config.conf`



## Επισκόπηση

### Controlling the simulation

Collecting Messages Statistics

Collecting Messages Statistics Examples

### Configuration of an Application

Passing parameters in the processor

Passing parameters Example

### JShawn

JShawn

### 3η Άσκηση

FloodMax



- ▶ Μέχρι τώρα για να εκτελέσουμε τις εξομοιώσεις γράφουμε όλες τις παραμέτρους σε ένα .conf αρχείο
- ▶ Αλλά τι γίνεται σε περίπτωση που θέλουμε να έχουμε πιο πολύπλοκο έλεγχο των εξομοιώσεων;
- ▶ Για παράδειγμα έστω ότι χρειαζόμαστε να εκτελέσουμε 100 φορές μια εξομείωση
  - ▶ for-loops
  - ▶ if-then
- ▶ Το JShawn είναι ένας διαφορετικός τρόπος για να γράφουμε config αρχεία που μας επιτρέπει πλήρη έλεγχο της εξομείωσης



- ▶ Οι κύριες εντολές ενός JShawn config αρχείου είναι:
  1. shawn.setGlobalVariable("variable","value") :  
Θέτει μια μεταβλητή, είναι ισοδύναμο με variable=value στο κλασικό config αρχείο
  2. shawn.runCommand("task-name", "extra parameters") :  
Εκτελεί ένα μια εντολή (task) με παραμέτρους
- ▶ Το JShawn υποστηρίζει πλήρη σύνταξη Java οπότε μπορείτε:
  - ▶ Να δηλώσετε μεταβλητές nx int l=0
  - ▶ Να τυπώσετε διαγνωσικά μηνύματα nx System.out.println("l = " + l)
  - ▶ Να χρησιμοποιήσετε for-loops nx for(int l=0;l<10;l++)



### Παράδειγμα JShawn αρχείου

```
shawn.runCommand('prepare_world', 'edge_model=simple
comm_model=disk_graph range=10');
```

```
shawn.runCommand('rect_world', 'width=25 height=25
count=800 processors=helloworld');
```

```
shawn.runCommand('simulation', 'max_iterations=10');
```



- ▶ Για να ξεκινήσετε μια εξομείωση με το JShawn δίνεται την εντολή από τον φάκελο buildfiles:

### Παράδειγμα JShawn αρχείου

```
java -jar ../jshawn-allinone.jar
-s ./shawn -b conf_file.jshawn
```



### Controlling the simulation

- Collecting Messages Statistics
- Collecting Messages Statistics Examples

### Configuration of an Application

- Passing parameters in the processor
- Passing parameters Example

JShawn

JShawn

### 3η Άσκηση

FloodMax



- ▶ Για την Υλοποίηση και πειραματική αξιολόγηση του αλγόριθμου FloodMax
- ▶ Για την υλοποίηση χρησιμοποιήστε το template που θα βρείτε στην σελίδα του μαθήματος
- ▶ Όλες οι μεταβλητές που θα χρειαστείτε βρίσκονται στο `floodmax_processor.h` και είναι
  - ▶ `bool leader_` : Αν ο κόμβος είναι ο leader
  - ▶ `int max_id_` : Η μέγιστη ταυτότητα που έχει λάβει ο κόμβος
  - ▶ `int diameter_` : Η διάμετρος του δικτύου (Θα την θέσετε στο config αρχείο για κάθε διαφορετική τοπολογία και θα την διαβάζει ο processor πριν ξεκινήσει ο αλγόριθμος)
- ▶ Χρειάζεται να τροποποιηθεί μόνο τις μεθόδους `boot()`, `work()` και `handle_flooding_message_node` του `floodmax_processor.cpp`



- ▶ Λεπτομέρειες υλοποίησης του αλγορίθμου FloodMax :
  1. Σε κάθε γύρο, οι processors στέλνουν ένα μήνυμα με την μέγιστη ταυτότητα σε όλους τους γείτονες
  2. Μόλις λάβουν μία ταυτότητα απο κάποιον γείτονα, την συγκρίνουν με την μέγιστη ταυτότητα:
    - ▶ Αν είναι μεγαλύτερη, θέτουν την μεταβλητή στην νέα τιμή
    - ▶ Αν είναι μικρότερη ή ίση την αγνοούν
  3. Μετά απο `diameter` γύρους:
    - ▶ Αν η μέγιστη ταυτότητα ισούται με την ταυτότητα της διεργασίας, ο processor μεταβαίνει στην κατάσταση εκλεγμένος
    - ▶ Κάθε processor τίθεται σε κατάσταση Inactive



- ▶ Πειραματική αξιολόγηση του αλγορίθμου FloodMax :
  - ▶ Θα εκτελέσεται 6 εξομοιώσεις του αλγορίθμου FloodMax
  - ▶ Για κάθε εξομοίωση θα χρησιμοποιήσετε διαφορετική τοπολογία και θα μετρήσετε τον αριθμό των μηνυμάτων που στάλθηκαν και των γύρων μεχρι τον τερματισμό της εξομοίωσης
  - ▶ Για κάθε μια εξομοίωση θα χρησιμοποιήσετε διαφορετική τοπολογία
    - ▶ `floodmax-topology-10-4.xml`: 10 κόμβοι διάμετρος 4
    - ▶ `floodmax-topology-100-22.xml`: 100 κόμβοι διάμετρος 22
    - ▶ `floodmax-topology-1000-30.xml`: 1000 κόμβοι διάμετρος 30
    - ▶ `floodmax-topology-2000-43.xml`: 2000 κόμβοι διάμετρος 43
    - ▶ `floodmax-topology-5000-58.xml`: 5000 κόμβοι διάμετρος 58
    - ▶ `floodmax-topology-10000-72.xml`: 10000 κόμβοι διάμετρος 72



- ▶ Ακολουθεί ένα παράδειγμα για το πως πρέπει να είναι το config αρχείο που θα χρειαστείτε:
- ▶ Προσοχή το range πρέπει να είναι 2!

### Παράδειγμα config αρχείου για την Άσκηση 3

```
prepare_world edge_model=list comm_model=disk_graph \  
transm_model=stats_chain range=2  
  
chain_transm_model name=reliable  
  
diameter=4  
load_world file=floodmax-topology-10-4.xml processors=floodmax  
simulation max_iterations=100
```

