On the Problem of Over-clustering in Tuple-based Coordination Systems

Matteo Casadei[1] Ronaldo Menezes[2] Robert Tolksdorf[3]
Mirko Viroli[1]

[1]Alma Mater Studiorum – Università di Bologna
{m.casadei,mirko.viroli}@unibo.it

[2]Florida Institute of Technology - Melbourne, USA
rmenezes@cs.it.edu

[3]Freie Universität Berlin, Institut für Informatik
tolk@inf.fu-berlin.de

SASO 2007
Work-In-Progress Track
Boston - USA
July 11, 2007

Introduction (1)

Scenario

- Coordination of active entities in open distributed scenarios by tuple-space systems
- Specific issue: tuple distribution and organization in a network of distributed tuple spaces

Why is tuple distribution and organization an important issue?

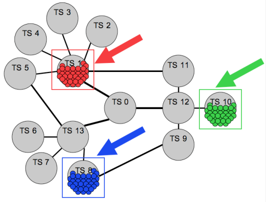
- It affects the scalability of a system ...
- ...indeed, if tuples are organized in a specific way, we can obtain a better system's scalability
- For instance, if similar tuples are kept close to each other it is easier for processes to find the specific tuple they need!

Introduction (2)

Tuple Organization: important issues

- Keeping similar tuples close to each other makes processes too dependent on aggregating nodes

- This leads to **over-clustering**: system's robustness is affected



Objectives

- Exploiting **self-organization** to build a tuple-space architecture with a scalable tuple-distribution mechanism
- Avoiding **over-clustering**

A Strategy to Organize Tuples

Developed by taking inspiration from ant's brood sorting

Aim of the Strategy

- Storing a tuple in a node with a high concentration of similar tuples
- Metaphor:
  - **out**s are **ants** carrying items and searching for a suitable node where to drop food
  - **tuples** are **items** of different kind carried by ants

How Tuple Distribution Works (0)

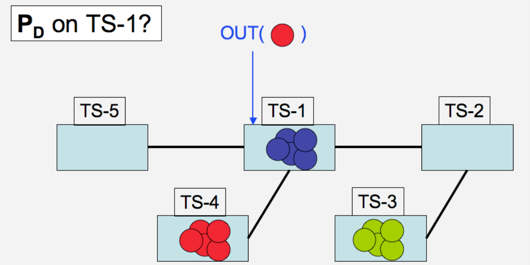
Phases

- Our tuple-distribution strategy is composed of **two phases**:
  - **Storing decision phase**: does a tuple have to be stored in the current node?
  - **Movement phase**: what is a good neighbor where to move a tuple?

Similarity Function

- How **similar** are two tuples *tu* and *te*?
  - $\delta(tu, te) = [0, 1]$

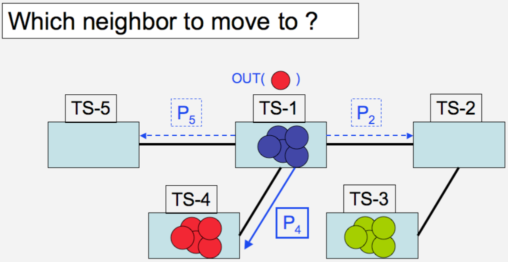
How Tuple Distribution Works (1)



Storing Decision Phase

- **Drop probability** $P_D = \left(\frac{F}{F+K}\right)^2$:
  - Concentration of tuples similar to *tu*: $F = \sum_{\forall t \in TS} \delta(tu, t)$
  - K is the number of hops left to an *out*-ant before storing the carried tuple *tu*
  - K is initially set to $Step_{MAX}$

How Tuple Distribution Works (2)

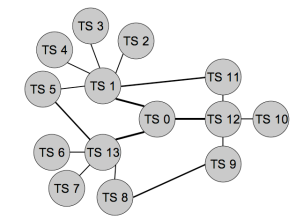


Movement Phase

- $\forall$ neighbor j, determine the **movement probability**
  $P_j = \frac{F_j}{\sum_{i=1}^{n} F_i}$
  - F_j: concentration of tuples similar to *tu* in tuple space j
  - $\sum_{i=1}^{n} F_i$: concentration of tuples similar to *tu* in the neighborhood of the current tuple space

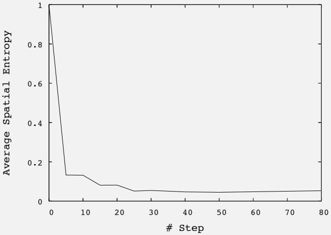
Simulation Results

Test Instance



- 14 tuple spaces
- Tuples belonging to 4 different templates
- Insertion of 200 tuples per template
- $\delta(tu, te) = \begin{cases} 1 & template(tu) = template(te) \\ 0 & \text{otherwise} \end{cases}$

Results: Spatial Entropy

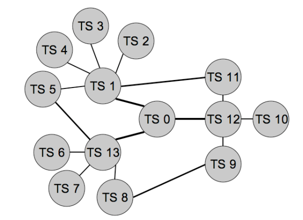


- 20 simulations per $Step_{MAX}$ value

Almost complete clustering!

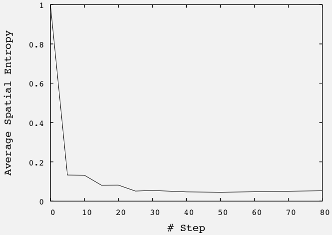
Simulation Results

Test Instance



- 14 tuple spaces
- Tuples belonging to 4 different templates
- Insertion of 200 tuples per template
- $\delta(tu, te) = \begin{cases} 1 & template(tu) = template(te) \\ 0 & \text{otherwise} \end{cases}$

Results: Spatial Entropy



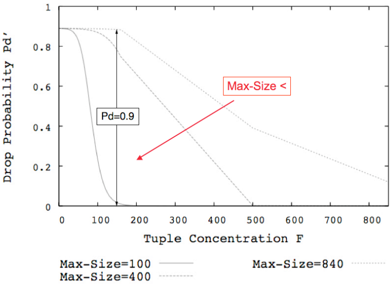
OVER-CLUSTERING!

A Solution to Over-Clustering

Max-Size

- For each node and tuple template, introduction of a *Max-Size* parameter in order to:
  - Regulate the maximum size of the clusters in the network by . . .
  - . . . modifying P_D when a cluster gets as big as *Max-Size*

How Max-Size Works



- A **sigmoid function** affects P_D
- The smaller *Max-Size* the higher the slope
  - Fixing a value of concentration F, the smaller *Max-Size* the smaller P_D

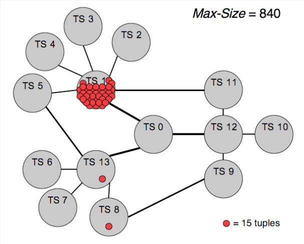
ANTI OVER-CLUSTERING!

Anti-Over-Clustering Simulation

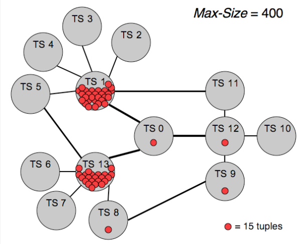
- Same test instance
- Insertion of tuples belonging to one template
  - Occurrence of 60 *out* operations per tuple space: 840 tuples, in total

Anti-Over-Clustering Simulation: Results

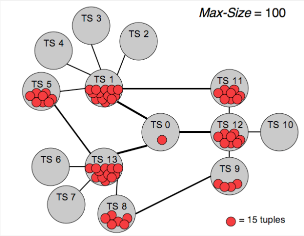
Tuple Organization in the Network



← From quasi-complete clustering . . .



. . . to clusters of smaller sizes for smaller values of *Max-Size* ↓



Conclusion and Future Work

Conclusion

- Developed a self-organizing strategy to tuple-distribution problem
  - Made some experiments with larger scale-free networks (100-1000 nodes)
- Found a first solution to avoid over-clustering

Future Work

- Experimenting with dynamic similarity functions to solve over-clustering with no need of a *Max-Size* parameter
- Devising a tuple-retrivial strategy based on what we have gotten regarding tuple-distribution

Thank you for your time!