Context-Dependent Coordination of Mobile Internet Agents

Franco Zambonelli

DSI - University of Modena and Reggio Emilia
franco.zambonelli@unimo.it
http://www.dsi.unimo.it/Zambonelli

Outline

- Internet agents
- Mobility
- Context-dependent Coordination
- Context-dependent Coordination in MARS
- Applications and Case Study
- Related (?) Work
- Conclusions and Future Work

Internet Agents

- Autonomous (*goal-oriented*)
  - execute in *autonomy* to achieve a *goal*
  - active objects with internal execution, event-handling and exception-handling capabilities, sometimes "intelligent"
- Interactive
  - inter-agent interactions
  - agent-resource interactions
- Network-aware (*location-aware*)
  - conscious about the *distribution* of the execution scenario
  - perceive the Internet as a set of local *execution environment* (e.g., Internet nodes and local administrative domains)
- Mobile?

Logical vs. Physical Agent Mobility

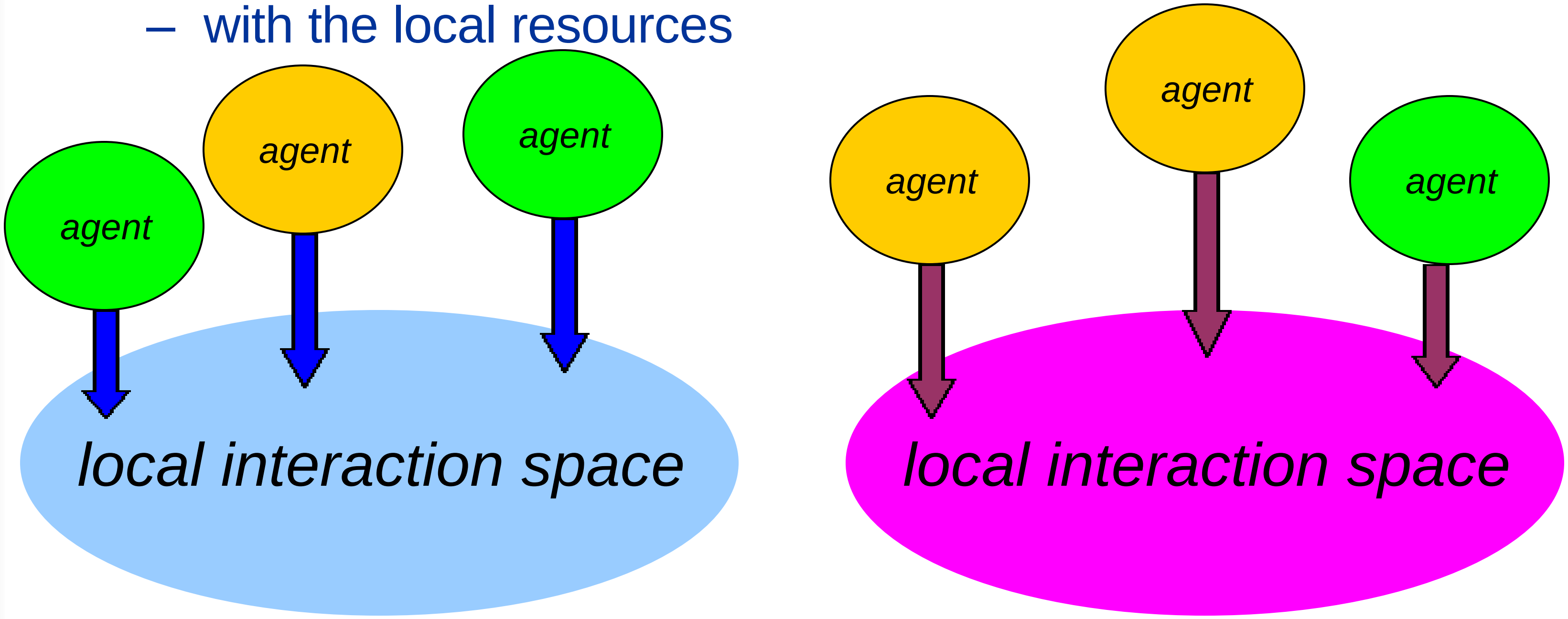
- Internet agents are mobile at a logical level
  - when they access resources in a network-aware fashion, they explicitly enter and interact in a different *execution environment*
  - agents move to different *contexts* while executing, even if code and state are fixed
  - we too "surf" the Internet from our desks ;-)
- Internet agents can be mobile at a physical level
  - *migration* of code, state, and execution threads
  - save of bandwidth (sometimes), reliability
  - natural mapping of logical into physical mobility
  - natural support for mobile computing
- Mobile devices are physically mobile agents

Local vs. Global Coordination

- Which coordination model for Internet agents?
- Global coordination
  - uncontrolled explosion of communication channels (spaghetti communications)
  - requires complex middleware for physically mobile agents
  - does not match the agents’ perception of the world
  - unsuitable to model mobile and ubiquitous computing
- Local Coordination
  - multiplicity of independent local interaction spaces (e.g., blackboards, meeting points, tuple spaces)
  - local interaction spaces accessed explicitly in a network aware fashion (e.g., via a URL) or implicitly
  - matches agent mobility and network-awareness

Coordination based on Local Interaction Spaces

- A multiplicity of interaction spaces, associated to a site or to a local domain
  - blackboards, tuple spaces, event-channels, etc...
- All interactions occur via the local interaction spaces
  - with the local agents
  - with the local resources



What is Context-Dependency?

- Context-dependency is *intrinsic* in mobility!
- Due to the very fact that agents move at different places and interact via different local interaction spaces:
  - they will meet different agents, with different attitudes and things to say
  - they will meet different resources….
- Example:
  - an *in(tuple)* operation on a tuple space has different effects depending on WHERE it s executed

…and more...

- Environment-dependent Coordination
  - The environment may be in need of somehow orchestrating/constraining the coordination activities of the local-executing agents
    - security reasons
    - resource control
    - facilitating access
- Application-dependent Coordination
  - application agents that are part of a given multi-agent system may have specific coordination needs/laws, to be enacted wherever they are
    - specification of complex protocol
    - control of opportunistic behaviors
- Require an **active role** of the interaction space!

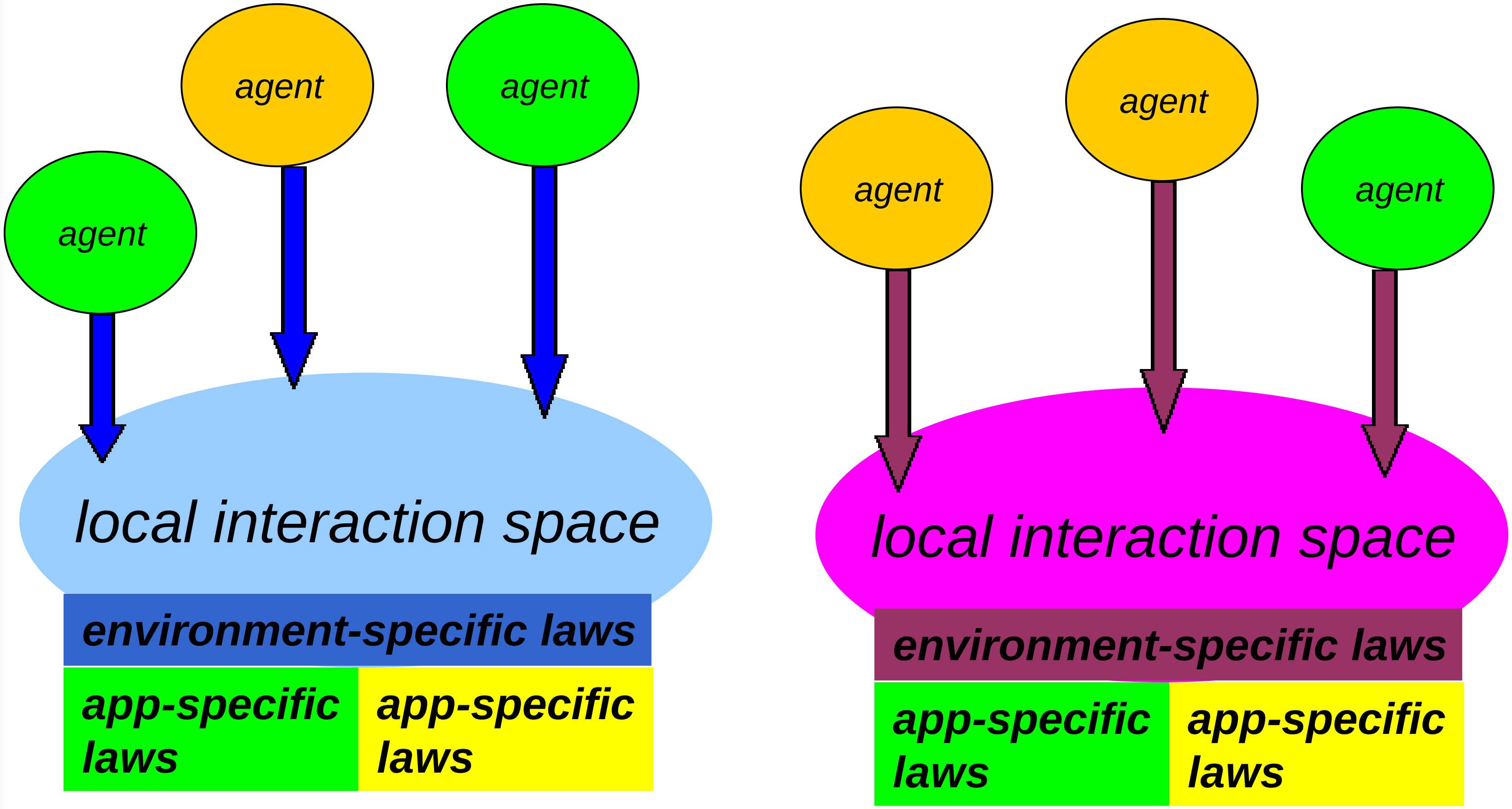
Programmable Interaction Spaces

- Interaction spaces can be made reactive and programmable
  - capability of monitoring all interaction events
  - capability of associating any computable reaction to any/specific class of interaction events
- Example: tuple spaces
  - detecting WHO invokes WHAT operation with WHICH parameter tuple
  - associating to this access (or to classes of accesses) reactions different from the standard pattern-matching
- the coordination activities can then be made subject to laws applying to the context in which they occur
  - ***application context*** and/or ***local environment*** context

Environment- vs. Application-Dependent Coordination

- Environment-dependent Coordination
  - programmability of the interaction space enables to associate specific computational activities to the accesses performed by there executing agents
  - possibly by discriminating on performed operations and agents’ identities
  - different environments = different, environment-specific, coordination laws
- Application-dependent Coordination
  - programmability of the interaction spaces can be used to let agents program the visited sites
  - programmability should influence only the same application context
  - different applications = different, application-specific, coordination laws

Context-Dependent Coordination: the Scenario



agent

agent

agent

local interaction space

environment-specific laws

app-specific laws

app-specific laws

agent

agent

agent

local interaction space

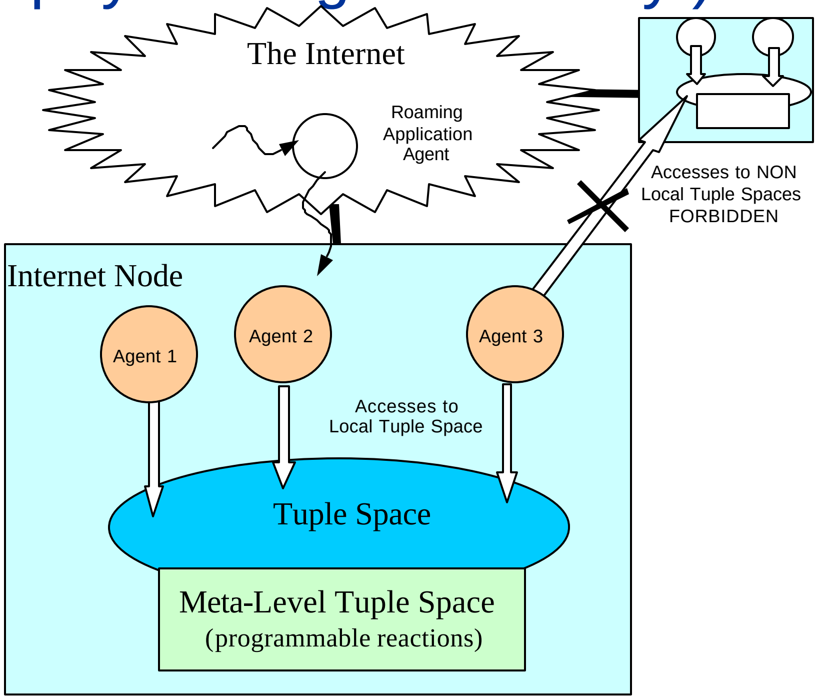
environment-specific laws

app-specific laws

app-specific laws

MARS: Mobile Agent Reactive Spaces

- available for different agent systems and compliant with JavaSpaces
- One tuple space on each node, implicitly accessed by local agents only (requires physical agent mobility!)
- Total mediation
  - inter-agent coordination
  - access to local resources
- Meta-level tuple space for programming tuple space behaviour



MARS: The Interface

- Object-oriented Linda operations
- Extended JavaSpaces' interface
  - *`Read/Readall (Entry tmp, Tr txn, long lease)`* reads one/all matching tuple(s)
  - *`Take/Takeall (Entry tmp, Tr txn, long lease)`* extracts one/all matching tuple(s)
  - *`Write (Entry e, Tr txn, long timeout)`* writes a tuple

  - `txn` can specify a transactions (*all-or-nothing* semantics)
  - *`lease`* blocking time for operations
  - *`timeout`* time to live for tuples
- Tuples are objects too

MARS: Programmability

- Meta-tuples to associate reactions to access events (meta-matching)
  `(Ag-Id, Tuple, OpType, ReactObj)`
- Access the base-level tuple space triggers pattern-matching in the meta-level to look for reactions to execute
- ***`ReactObj`***: Java object with a single method (the reaction itself)
- Example

  `(ag@mo.it, null, read, O1)`

  *triggers the reaction method of object **`O1`** when the agent **`ag@mo.it`** performs a **`read`** operation*

MARS: Examples of Reactions

```
// can be associated to readAll operations on FileEntry templates with HTML extension
//via the meta-level tuple:   (HTML2HTM_instance, (null, “html”, null, null), readAll, null)

class HTML2HTM implements Reactivity
{
public Entry[ ] reaction(Space s, Entry Fe, Operation Op, Identity Id)
// no match already occurred if the site has only htm files
{ Fe.Extension = “htm”;                              // modifies the extension of the required files
  return s.readAll(Fe, null, NO_WAIT);               // read FileEntry tuples with HTM extension
}}
_______________________________________________________________________________
// can be associated to take operations via: (TransformTakeObj, null, take, null)

class TransformTake implements Reactivity
{
public Entry reaction(Space s, Entry Fe, Operation Op, Identity Id)
{ if(matched(Fe)) { // if a match has been produced
              if (Id.equals(manager)) // check for the identify of the agent performing the operation
                            return s.take(Fe, null, NO_WAIT);// the tuple is deleted from the spac
              else  {SecurityRegister.add(“takeAll”, Fe, Id);    // log the access
                            return Fe }  // the tuple is returned but it not extracted
else return null;             }              // no match has been produced   }}
```

Context-Dependent Coordination in MARS

- Environment-dependent Coordination
  - the local administrator can publish any resource in the local tuple space
  - specific reactions programmed in the tuple space can provide
    - different views of the same tuple space to different agents
    - dynamic access to services and to tuple space content
- Application-dependent Coordination
  - agents can dynamically install their own reactions in the tuple space of the visited sites to influence their application-specific coordination activities
  - the local tuple space ensure that these application-specific reactions apply only to the agents of the same application
- Composition and execution of reactions: pipeline

Exploiting Environment-Dependent Coordination

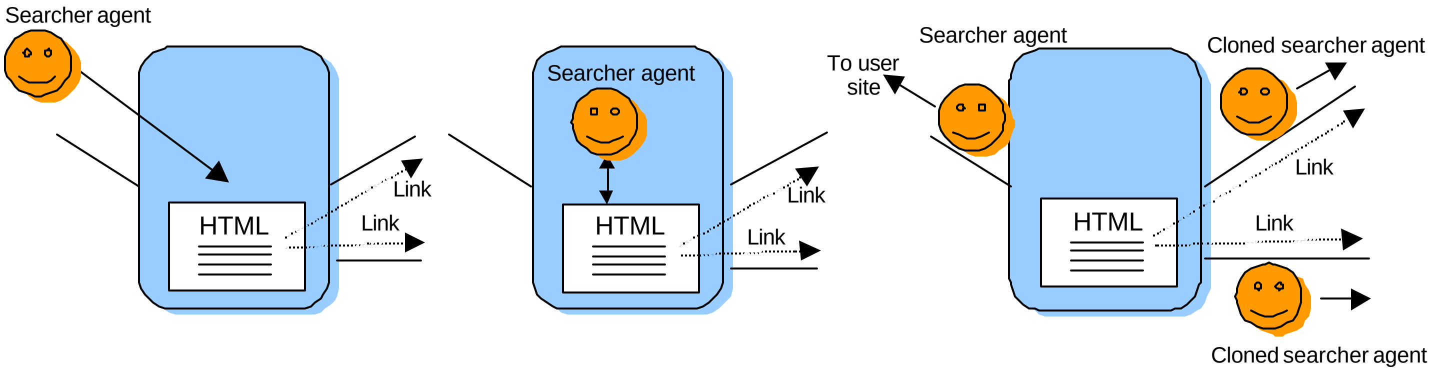
- Local interaction space as the gate via which to control local resources
  - monitoring policies
  - security policies programmed in the interaction space
  - resource control policies: stateful policies to control the amount of resources retrieved/exploited
- Facilitating access to local resources:
  - intelligent agents can handle heterogeneity and uncertainty about the environment
  - intelligent environments can help stupid agents in findings what they are looking for
- “Agoras” for open agent meetings
  - the environment fixes the rules for participating
  - and the rules for interacting in the meeting

Exploiting Application-Dependent Coordination

- Complex, multi-step, conversations
  - the interaction space can control the evolution of the protocol
  - can control the correct actions of the participants
  - can control the correct syntax of messages
- Implicit knowledge-exchanges
  - an agent has acquired new knowledge
  - wants to transmit it other agents
  - that can be done via a proper programming of the interaction space
- Application specific coordination laws
  - fixing the rules of interactions in a group of agents
  - enforced independently of the physical location of interactions
  - via a distributed programming of remote tuple spaces

Case Study: Distributed Information Retrieval

- Agents reach remote sites to access and analyze HTML pages
- Cloning to follow remote links (possibly interesting) ➩ dynamic tree of mobile agents
- After a visit, agents come back with the URLs of interests (*e.g.* matching a keyword)



Coordination in the Case Study

- With the local execution environment:
  - retrieve info on local HTML files
- Facts:
  - require access control
  - heterogeneity of execution environments
- Inter-agent
  - avoid duplicated work on the same sites (due to cross-references in HTML pages)
- Facts
  - agents are dynamically created
  - cannot foresee their number, their location, and their itinerant activities in the network

Environment-Dependent Coordination in the Case Study

- The local interaction space as a way to publish and access to local HTML files
- The behavior of the interaction space can be tuned to
  - integrate security and resource control
    - agents of class X can retrieve only 10 HTML files per day!
  - facilitate access
    - the site stores files with HTM extension
    - space programmed to produce HTML files on-demand
- In MARS:
  - tuples representing HTML files
  - a reaction makes “htm” tuples match with the “html” request of agents requesting HTML tuples

```
// associated to readAll operations on FileEntry templates
//  with HTML extension via the meta-level tuple:
(HTML2HTM_instance, (null, “html”, null, null), readAll, null)
class HTML2HTM implements Reactivity
{ public Entry[ ] reaction(Space s, Entry Fe, Operation Op, Identity Id)
// no match already occurred if the site has only htm files
{ Fe.Extension = “htm”;
  return s.readAll(Fe, null, NO_WAIT); }}
```

Application-Dependent Coordination in the Case Study

- One can exploit the local interaction space as a simple message repository
  - *message(I have already visited this site!)*
  - does not require programmability
- More complex inter-agent coordination rules
  - the next agent arriving on the site should have access only to those files that have been modified since the previous visit
  - an application-specific reaction can provide this behavior
- In MARS:
  - an will perceive only those HTML tuples whose file has been modified since the previous visit

```
//class IncrementalVisit implements Reactivity
{ private Date visit;
 public IncrementalVisit()  {  visit = new Date(); }
public Entry reaction(Space s, Entry Fe, Operation Op, Identity Id)
 {if (s.matched(Fe) && !(FileEntry)Fe.ModificationTime.before(visit))
                      { visit = new Date(); return Fe;  }
  else                return null;   } }
```

Other Application Areas (1)

- Agents’ Markets
  - local interaction spaces as abstractions of marketplaces
  - environment-dependent coordination to embed the laws to govern the local negotiation activities
  - In MARS, implementation of local interaction spaces implementing different kinds of auctions
    - same interface
    - different rules
    - control of self-interested behaviors (e.g., bidders’ collusions)
- Agent-based Workflow and Cooperative Work
  - local interaction spaces as abstractions of workplaces
  - programmability exploited to govern workflow rules
    - who can modify which document and when

Other Application Areas (2)

- Ubiquitous and Wearable Computing
  - devices embedded in an environment (an hospital room, a kitchen, a cloth)
  - the device manufacturer should not be obliged to configure each and every device to be adapted to the environment in which it will be embedded
  - instead, the environment should provide laws to enable and rule the coordination of the embedded devices
- Mobile Computing
  - we move, carrying on (or wearing) computing facilities
  - we change the context in which our computing facilities operate
  - there must be an active role of the context to support effective interaction
  - need of supporting *context-awareness*

Impact on Design and Development (1)

- Context-dependent approach to design:
  - intra-agent engineering:
    - what should each agent do?
    - how should it do it?
  - inter-agent engineering
    - how should my agent coordinate?
    - which information should they exchange and how?
  - site/environment engineering (for administrators):
    - how should I set up my environment for hosting agents safely and effectively?
- Clear separation of concerns
- Design for change perspective

Impact on Design and Development (2)

- Exploiting programmable interaction spaces during development:
  - write the agent code
  - write the code for programming the coordination policy to be inserted/downloaded into the interaction space
- Ease of maintenance/upgrade
  - no side effects
  - changes to agents’ code do not influence coordination code
  - vice versa
- Still free of neglecting programmability of interaction spaces and of re-using old code/applications!

Impact on Design and Development: the Case Study

- Intra-agent engineering:
  - how should my agents extract, elaborate, and render the retrieved information?
  - how should my agents plan their itinerary?
- Inter-agent engineering
  - how should my agents avoid multiple visits on a site?
  - what relevant information should they exchange with each other?
- Environment engineering:
  - how should I facilitate agents' work?
  - how should I avoid them to access private information or consume too many resources?

Related Coordination Models

- TuCSoN, Joint project with MARS
  - logic tuple spaces with well-founded model of programmable behaviour
  - Local + remote interaction
    - does not require physical agent mobility
    - ease of composition of coordination laws
- Lime, Univ. Washington in Sant Louis
  - unifying model for devices, users, and agents
  - Interactions via a local (carried on) tuple space
  - Merging of tuple spaces in case of “meetings”
    - coordination activities implicitly influenced by merging rules
- LGI, Rutgers University
  - programmable – security oriented model – for programmable coordination via peer-to-peer interactions

A Possible Criticism

- That’s not the way the real world works….
  - the environment is a passive entity
  - everything is active is indeed an agent
  - coordination is an internal activity of an agent, which relates it to the external world
  - agents modify the environment and behaviours emerges only by the fact that agents communicate with each other and modify the environment
  - global interactions exist and are relevant
- That can be true, but….
  - Is this the good abstraction to model complex systems?
  - Is this the good abstraction to build complex multi-agent system?
  - E.G., in the Internet, should we consider all servers as agents or as part of the environment?

A More General Perspective on Context-Dependent Coordination

- Context-dependent coordination modelling is pervasive
  - Human societies and social conventions
  - Ant-colonies and ant-inspired multi-agent systems
  - Cellular Automata
  - Distributed Systems Management
  - Open Physical Systems

Social Behaviors and Social Conventions

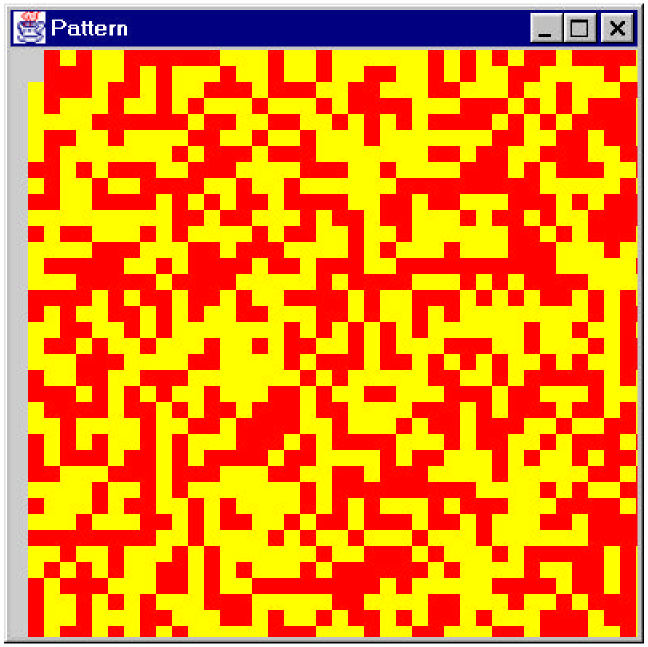
- Human societies lives in a context-dependent coordination world
  - the same group will behave differently at a gala party and at a pub (environment-dependent)
  - the same party in the same place will be different depending on which group of people is in there (application-dependent)
- It may be true that everything is in our minds, not in the environment. However:
  - we “feel” the influence of the environment or of the group/application
- Organisational and Social Approaches to Multi-agent Systems Development
  - roles to be played by agents in the context of an organisational structure
  - organisational rules and social laws to be respected

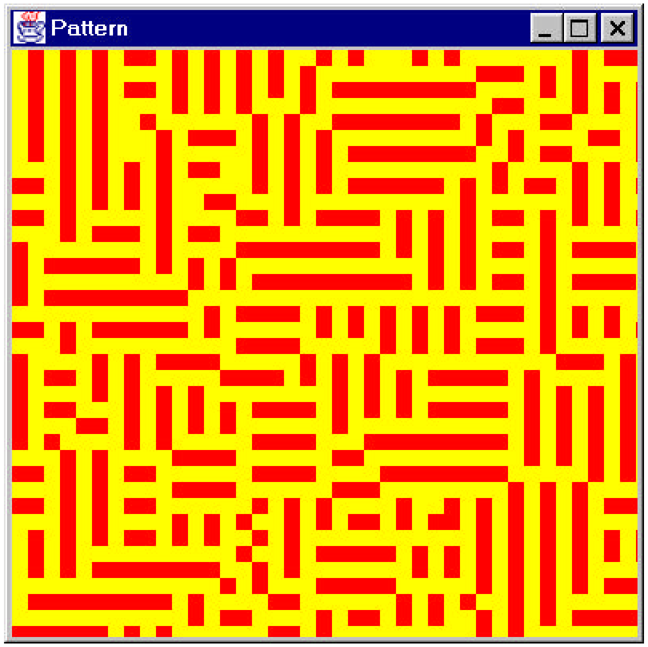
Ant-based Routing

- Very simple “mobile agents” explore the environment and obey to very simple rules
  - they sense “pheromones” in the environment
  - depending on gradient of pheromones density:
    - move in a given direction
    - put new pheromones in the environment
- Emergence of patterns:
  - minimal path
  - fault-tolerant and adaptive routing
- Other applications areas (load balancing, sorting, …)
- The **environment must guarantee specific laws** for the rate of pheromones evaporation!
- The laws of the environment determine the global behaviour!

Cellular Automata

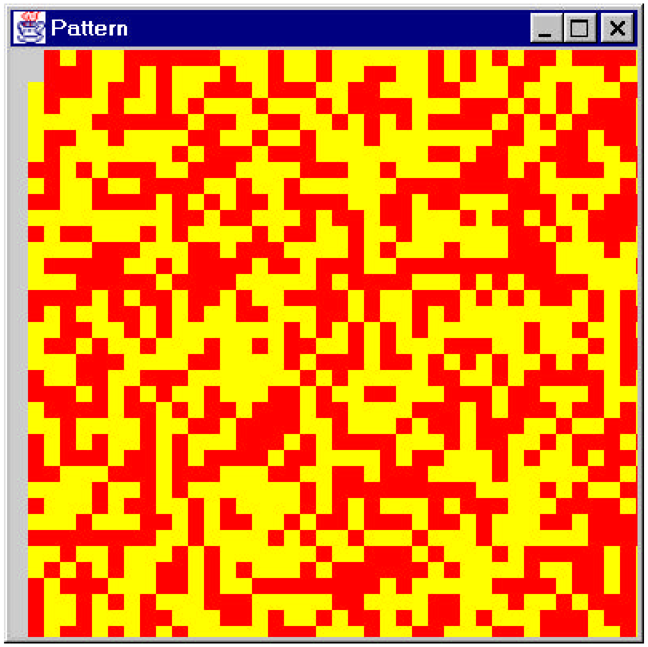
- In “normal”, close, 2D-CA, no emergence of long-range interactions and symmetry breaking
- Local interactions --> local effects

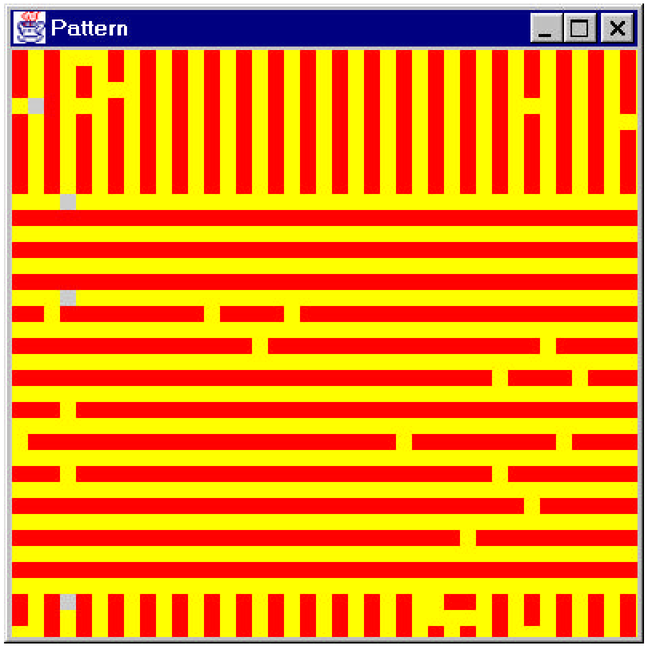




Cellular Automata

- In the presence of an open environment, emergence of long-range interactions and symmetry breaking
- The ***laws of the environment*** determines the behaviour of the agents!
- Similar to Prigogine's dissipative structures…..





Open Distributed Systems

- Managing open distributed systems requires peculiar approaches

- Example: load balancing in distributed Web-servers
- Traditional LB policies are inadequate
- Specific LB policies must take into account that
  - not all the workload is under direct control of the policy (openness of the system)
  - the distribution of arrivals can be various (poisson vs. heavy-tailed)
- Strong inter-play between application-dependent issues and environment-dependent ones
  - need to coordinate resource assignments
  - taking into account the dynamics of the environments

Open Physical Systems

- Describing the behaviour of open physical systems at a macroscopic level requires:
  - definition of boundaries
  - definition of energy flows between boundaries and internals
  - local interactions between internal components
- Again, the inter-play between the local interaction laws and the laws of the environment determines the global behaviour of the system
- Complex behaviours can emerge:
  - Bènard Cells, Chemical Clocks, etc.

Open Issues

- Security issues
  - Internet asks for secure coordination models
  - tight relation between coordination and security
- Software engineering issues
  - methodologies for shaping the environment other than for shaping agents
  - trade-off between agent-centered and coordination-centered design
  - dynamics of emergent behaviours
  - formal models, experiences, and simulation tools
  - towards a general theory of context-dependent coordination?
- Internet standards, of course

Conclusions

- Context-Dependent Coordination
  - mobile and autonomous application components (agents)
  - local interactions in local environments
  - enacting of application-dependent coordination laws
  - constrained by environment-dependent coordination laws
- Prototype infrastructures already available
  - implicitly defining a context-dependent coordination model
  - suitable for the development of Internet-agent applications
- Several open issues
  - methodologies for development and control
  - relation with other approaches and disciplines