

## Interaction, Direct and Indirect

David Keil  
CSETS theory seminar, October 5, 2005  
Prof. Aggelos Kiyias, University of Connecticut

David Keil Interaction, Direct and Indirect 9/05 1

## Fundamental Questions Underlying Theory of Computation

**What is computation?  
How do we model it?**

**A:**

- Algorithmic
- Interactive
  - Sequential
  - Multiagent

David Keil Interaction, Direct and Indirect 9/05 2

## Algorithms vs. interaction

|   |   |
|---|---|
| <p><b>Algorithmic computation:</b><br/>finite transformation of input to output</p> <p><i>input:</i> finite-size (string or number)</p> <p><i>closed system:</i> all input available at start, all output generated at end</p> <p><i>behavior:</i> functions, algorithmic transformation of input data to output data</p> | <p><b>Interactive computation:</b> ongoing process which performs a task or delivers a service</p> <p><i>dynamically</i> generated stream of input tokens (requests, percepts, messages)</p> <p><i>open system:</i> later inputs depend on earlier outputs and vice versa (I/O entanglement, history dependence)</p> <p><i>behavior:</i> processes, components, control devices, reactive systems, intelligent agents</p> |
|---|---|

David Keil Interaction, Direct and Indirect 9/05 3

## Modeling Interactive Computation

- **Many interactive models**
  - Reactive [MP] and embedded systems
  - Dataflow, I/O automata [Lynch], synchronous languages, finite/pushdown automata over infinite words
  - Interaction games [Abramsky], online algorithms [Albers]
  - TM extensions: on-line Turing machines [Fischer], interactive Turing machines [Goldreich]
- **Concurrency Theory**
  - Focuses on *communication* (between concurrent agents/processes) rather than computation [Milner]
  - *Orthogonal* to the theory of computation and TMs.
- **What makes our approach unique?**
  - Communication (I/O) is part of computation.
    - a paradigm change
  - Bridging the gap between concurrency theory (labeled transition systems) and traditional TOC.
    - models borrow from both fields
  - Focus on expressiveness
    - rather than mere convenience of modeling

David Keil Interaction, Direct and Indirect 9/05 4

## Outline

- A. Single-agent interaction
  1. A minimal extension of TMs
  2. Expressiveness results
- B. Multiagent interaction
  1. Direct and indirect interaction
  2. Dynamic persistent environments
  3. Modeling MAI

David Keil Interaction, Direct and Indirect 9/05 5

## Overview: PTMs

- A new way of interpreting Turing-machine computation, based on dynamic stream semantics
- A PTM is a Turing machine that has an additional, persistent worktape
- The class of PTMs is isomorphic to a general class of effective transition systems
- PTMs without persistence (amnesic PTMs) are less expressive than PTMs in general.
- An analogue of the Church-Turing hypothesis: PTMs capture sequential interactive computation
- *Reference:* (Goldin et al, 2004)

David Keil Interaction, Direct and Indirect 9/05 6

### Three-tape Turing Machines (N3TM)

$s$  - current state  
 $w_1$  - contents of input tape  
 $w_2$  - contents of work tape  
 $w_3$  - contents of output tape  
 $n_1, n_2, n_3$  - tape head posns

- N3TM configurations:  $\langle s, w_1, w_2, w_3, n_1, n_2, n_3 \rangle$
- Computation = a sequence of transitions between configurations, from initial to halting.

David Keil Interaction, Direct and Indirect 9/05 7

### Extending N3TM Computations

- Consider iterating TM computations without erasing worktape
- Dynamic stream semantics**
  - Inputs are streams of dynamically generated tokens (strings).
  - For each input token, there is an N3TM macrostep generating the corresponding output token.
- Persistence (memory)**
  - The contents  $w$  of the work tape at the beginning of each macrostep is the same as at the end of the previous one.

David Keil Interaction, Direct and Indirect 9/05 8

### Persistent Turing Machine (PTM)

**PTM: N3TM with persistent stream-based computational semantics**

- Persistent Stream Language** of a PTM: set of streams  $\{ \langle in_1, out_1 \rangle, \langle in_2, out_2 \rangle, \dots \}$
- Conductive stream semantics:  
 $S = (\Sigma^* \times \Sigma^*) \times S$

David Keil Interaction, Direct and Indirect 9/05 9

### Formal Definition

(Coinductive definition, relative to N3TM  $M$  and memory  $w$ )

$$PSL(M(w)) = \{ \langle (w_i, w_o), \sigma \rangle \in S \mid \exists w' \in \Sigma^* : \langle w_i, w \rangle \Rightarrow \langle w', w_o \rangle \wedge \sigma' \in PSL(M(w')) \}$$

$$PSL(M) = PSL(M(\varepsilon))$$

**PSL equivalence:**  $M_1 =_{PSL} M_2$

$$PSL = \{ PSL(M) \mid M \text{ is a PTM} \}$$

David Keil Interaction, Direct and Indirect 9/05 10

### PTM Example

- Answering Machine (AM)**
  - $f_{AM}(\text{record } Y, X) = (ok, XY)$
  - $f_{AM}(\text{erase}, X) = (\text{done}, \varepsilon)$
  - $f_{AM}(\text{playback}, X) = (X, X)$
- PSL(AM) contains:**
  - (record **hello**, ok), (erase, done),
  - (record **Farhad**, ok), (record **Arbab**, ok),
  - (playback, **Farhad Arbab**), ...

David Keil Interaction, Direct and Indirect 9/05 11

### PTM Example: Latch

- At each step, output first bit of previous step.**
  - inputs  $in_1$ ; outputs 1
  - inputs  $in_2$ ; outputs 1<sup>st</sup> bit of  $in_1$
  - inputs  $in_3$ ; outputs 1<sup>st</sup> bit of  $in_2$
  - ...
- PSL(Latch) contains:**
  - $\{(1,1), (0,1), (0,0), (1,0), \dots\}$
- PTM as a Labeled Transition System**
  - Latch has 3 states, meaning "contents of worktape"
  - The labels are input/output pairs, as in the interaction stream.

David Keil Interaction, Direct and Indirect 9/05 12

### Interactive Transition Systems over $\Sigma$

$\langle S, m, r \rangle$

- $S$  is set of states
- $r$  is initial state (root)
- $m$  is transition relation

$$m \subseteq S \times \Sigma^* \times S \times \Sigma^*$$

Required to be recursively enumerable

David Keil Interaction, Direct and Indirect 9/05 13

### Interactive Stream Equivalence

- Infinite sequences of input/output token-pairs emanating from a particular ITS state
- $ISL(T)$ , the *interactive stream language* of an ITS  $T$ , is the set of all such sequences emanating from  $T$ 's root.

$T_1 =_{ISL} T_2 \text{ if } ISL(T_1) = ISL(T_2)$

David Keil Interaction, Direct and Indirect 9/05 14

### Outline

- A. Single-agent interaction
  1. A minimal extension of TMs
  2. Expressiveness results
- B. Multiagent interaction
  1. Direct and indirect interaction
  2. Dynamic persistent environments
  3. Modeling MAI

David Keil Interaction, Direct and Indirect 9/05 15

### Amnesic PTMs:

"half-way" between TMs and PTMs

- Amnesic PTMs extend TMs with stream-based semantics. *At least as expressive as TMs*
- Unlike PTMs, they lack persistence.

**Example: squaring machine** ( $out_i = in_i^2$ )  
 [Prasse & Rittgen]

David Keil Interaction, Direct and Indirect 9/05 16

### Amnesic PTM Computation:

stream-based but not persistent

$$ASL(M) = \{ \langle w_i, w_o \rangle, \sigma' \in S \mid \exists w \in \Sigma^* : \langle w_i, \epsilon \rangle \Rightarrow \langle w', w_o \rangle \wedge \sigma' \in PSL(M(w')) \}$$

**ASL equivalence:**  $M_1 =_{ASL} M_2$

$$\mathcal{ASL} = \{ ASL(M) \mid M \text{ is a PTM} \}$$

**PTM  $M$  is amnesic if  $PSL(M) \in \mathcal{ASL}$**

David Keil Interaction, Direct and Indirect 9/05 17

### Summary of Results [I&C'04]

David Keil Interaction, Direct and Indirect 9/05 18

### Sequential Interaction

- **Sequential interactive computation:**  
*system continuously interacts with its environment by alternately accepting an input string and computing a corresponding output string.*
- **Examples:**
  - method invocations of an object instance in an OO language
  - a C function with static variables
  - queries/updates to single-user databases
  - recurrent neural networks
  - control systems
  - online computation
  - transducers
  - dynamic algorithms
  - embedded systems

David Keil Interaction, Direct and Indirect 9/05 19

### Sequential Interaction Thesis

Whenever there is an effective method for performing sequential interactive computation, this computation can be performed by a PTM

- **Universal PTM:** simulates any other PTM
  - Need additional input describing the PTM
- **Example:** simulating Answering Machine  
 (simulate **AM**, will-do),  
 (record **hello**, ok), (erase, done), (record **Farhad**, ok),  
 (record **Arbab**, ok), (playback, **Farhad Arbab**), ...

*Simulation of other sequential interactive systems is analogous.*

David Keil Interaction, Direct and Indirect 9/05 20

### Outline

- A. Single-agent interaction
  1. A minimal extension of TMs
  2. Expressiveness results
- B. Multiagent interaction
  1. **Direct and indirect interaction**
  2. Dynamic persistent environments
  3. Modeling MAI

David Keil Interaction, Direct and Indirect 9/05 21

### Overview : Multiagent interaction

- Multiagent interaction involves both direct (targeted) and indirect (anonymous) communication.
- Current modeling of concurrency uses a strictly message-passing notion of interaction.
- In dynamic persistent environments, indirect interaction may occur
- It is essential to the power of multiagent systems
- New models are required that explicitly represent indirect interaction via the environment.
- *Reference:* (Keil and Goldin, 2005)

David Keil Interaction, Direct and Indirect 9/05 22

### Multistream interaction (MSI)

- MSI is distinguished from *sequential interaction*
- MSI is characterized by *multiple concurrent streams* of input/output, and by *creation or destruction* of these streams during a computation

The diagram illustrates Multistream Interaction (MSI). It shows a central group of five square agents enclosed in a dashed box. Arrows represent stream inputs and outputs. Some arrows point into the group from the left, while others point outwards to the right. There are also arrows between the agents themselves, indicating internal interactions. A legend on the left shows a square for 'Agents' and an arrow for 'Stream inputs and outputs'.

David Keil Interaction, Direct and Indirect 9/05 23

### Direct and indirect interaction

- *Direct interaction* is interaction via messages; the identifier of the recipient is specified in the message
- *Indirect interaction* is interaction via persistent, observable state changes in a common environment; recipients are any agents that will observe these changes
- Examples of indirect interaction: Stigmergy, distributed computing

David Keil Interaction, Direct and Indirect 9/05 24

### Stigmergy: Indirect interaction in natural settings

#### 1. Termites gathering chips

Move at random, pick up chip when encountered, put down when another found

#### 2. Ants foraging for food

Ants leave chemical trail, prefer existing trails, blaze shorter and shorter trails to and from food

#### 3. Slime mold dividing and aggregating

These amoeba may aggregate by emitting a chemical, migrating toward its greatest concentration

### Properties of indirect interaction

- **anonymity** (recipient ID not used in access)
- **time decoupling** (state changes persist)
- **space decoupling** (agents need not meet)
- **locality** (interaction is between agents that visit the same data location)
- **non-intentionality** (agents need not have goal of communicating)

### Outline

- A. Single-agent interaction
  1. A minimal extension of TMs
  2. Expressiveness results
- B. Multiagent interaction
  1. Direct and indirect interaction
  2. **Persistent environments**
  3. Modeling MAI

### Persistent environments

An environment is *persistent* w.r.t. an agent if its outputs to the agent depend on the agent's earlier actions within it.

An environment is *amnesic* if it is not persistent.

*Examples:*

- A notebook (persistent w.r.t. a user)
- A sports event as televised (dynamic and amnesic)

### Persistence and MASs

- A persistent environment, w.r.t. an agent  $A$ , is a set of agents or shared variables that  $A$  observes to change as a whole, in response to  $A$
- Since this environment has persistent state, how it responds to  $A$  may evolve over time
- All nontrivial (coherent) MASs have components that are persistent environments of other components

### Multiagent systems require indirect interaction

**Theorem:**

- Let  $S$  be a multiagent system that contains some agents (and possibly shared variables) that form a persistent environment w.r.t. the other agents.
- Then indirect interaction occurs in  $S$ .

*Significance:* Since indirect interaction is an essential feature of multiagent systems, a model of MAI must incorporate this form of interaction

## Outline

- A. Single-agent interaction
  1. A minimal extension of TMs
  2. Expressiveness results
- B. Multiagent interaction
  1. Direct and indirect interaction
  2. Persistent environments
  3. **Modeling MAI**

## Modeling multiagent interaction

- R. Milner's CCS model represents interaction as *binary communication* via *message passing*
- Shared memory is modeled as *processes*
- $\pi$ -Calculus models *mobility* (linking-unlinking), but still as targeted send/receive

## Limits of models based on message-passing

- The *semantics* of MASs include anonymous interaction among aggregates and via the environment
- The environment may be passive, non-agent-like
- Mobility, time decoupling, name decoupling imply a different semantics from targeted send/receive

## References

- [GK03] D. Goldin, D. Keil. Modeling indirect interaction in open computational systems. *Proceedings, TAPOCS 2003*.
- [GSAS04] D. Goldin, S. Smolka, P. Attie, E. Sonderegger. Turing machines, transition systems, and interaction. *Information and Computation Journal*, 194(2):101-128, 2004.
- [KG05] David Keil, Dina Goldin. Adaptation and evolution in dynamic persistent environments. Presented at FInCo2005, Edinburgh, April 2005.
- [Mil93] R. Milner. Elements of Interaction. *CACM* 36(1): 78-89, 1993.

[Additional material on No Free Lunch]

## Fitness and optimization

- Assume we are working with utility-based agents
- Let  $f$  be a function that describes reward of an agent in a static environment, given the agent's actions
- Then *optimizing* the agent means finding actions that maximize  $f$
- It would be nice if a good general-purpose algorithm existed for optimizing reward functions
- Evolution of species in a stable habitat is function optimization

### The function-optimization problem

- Let  $f: \mathbf{N}^k \rightarrow \mathbf{R}$  for some  $k$  (the *arity* of  $f$ )
- *Problem*: Find some  $x \in \mathbf{N}^k$  s.t.  $f(x)$  is maximal
- *Example*: Suppose  $x$  is the set of proportions of ingredients in a fuel mixture,  $f(x)$  is fuel efficiency under this mixture
- *Optimizing*  $f(x)$  means finding the most efficient mixture
- For an *algorithm* to optimize a function we must have  $f: X \rightarrow Y$  with  $X, Y$  finite

### No Free Lunch

- *Theorem*: No function-optimization algorithm performs better in the general case than random choice [WM95]

$$\sum_f P(\bar{c} \mid f, m, a_1) = \sum_f P(\bar{c} \mid f, m, a_2)$$

- *Implication*: Function optimization usually requires special knowledge of the domain, i.e., heuristics

### Adaptation

- Consider an environment in which agents may receive *reward* inputs, and that can change as a result of actions by agents
- Then to increase reward, those agents may *adapt* to the environment, and may *change* the environment to benefit themselves
- Those environments, DPEs, are the ones that pose the problem of *adaptation*

### The NFL paradox

- Whereas no general-purpose function optimizing algorithm exists, the process of evolution has provably yielded adaptive and even intelligent life, contradicting the NFLT
- NFLT assumes *static* environment, whereas evolution occurs in dynamic persistent environments
- Evolution is an *adaptive* process of interaction and *learning* from the environment

### Conjectures

- No NFLT exists for DPEs
- Certain interactive protocols obtain better results than others in optimizing inputs from dynamic persistent environments.
- *Examples*:
  - *Selection, recombination, and mutation of DNA* are a robust protocol for the generation of survivable life forms.
  - The *scientific method* provides better results than random guessing