Team Automata

and their Application to

Security Analysis

Maurice H. ter Beek

Institute of Information Science and Technologies

National Research Council

ISTI-CNR

Pisa, Italy

Wednesday 10 March 2004

IST-TUG

Outline

Background & Origins of Team Automata (TA)

Foundations of TA:

definitions, examples & results

Applying TA to Security Analysis:

EMSS protocol, insecure communication scenario & GNDC schema

Background of TA

Computer Supported Cooperative Work is:

? how do groups of people work together

? how can technology (groupware) assist

CSCW needs:

Conceptual level:
define concepts & terminology consistently

Architectural level:
describe & compare groupware systems

$\Rightarrow$ Ellis introduced idea of *team automata* for modeling both levels informally in

Team Automata for Groupware Systems

(GROUP'97 - International ACM Conference on Supporting Group Work, 1997)

Origins of TA

Theoretical source of inspiration:

Input/Output automata (IOA) – Lynch & Tuttle

Conceptual source of inspiration:

Collective Intelligence in Computer Based Collaboration – Smith

(Computers, Cognition and Work series, 1994)

"well-structured groups [i.e. teams] outperform individuals in certain tasks" – but: also calls for "models capturing concepts of group behavior"

My Ph.D. Thesis *Team Automata* shows:

usefulness of TA not limited to modeling groupware notions & systems, but:

extends to modeling collaboration in reactive, distributed systems in general !

Foundations of TA

[BEKR03] ter Beek, Ellis, Kleijn & Rozenberg

Synchronizations in Team Automata for Groupware Systems

(CSCW—the Journal of Collaborative Computing, 2003)

<u>TA:</u>

- model logical architecture of a design
- abstract from concrete data & actions
- describe behavior in terms of
  - state-action diagram (automaton)
  - role of actions (input, output, internal)
  - synchronizations (simultaneous execution of shared actions)
- crux: automata composition

Component Automaton

$$\mathcal{C} = (Q, (\Sigma_{inp}, \Sigma_{out}, \Sigma_{int}), \delta, I)$$

Q set of *states*

$\Sigma = \Sigma_{inp} \cup \Sigma_{out} \cup \Sigma_{int}$ *alphabet* (partition)

$\delta \subseteq Q \times \Sigma \times Q$ *transition relation* $\quad q \xrightarrow{a} q'$

$I \subseteq Q$ set of *initial states*

$\left.\begin{array}{l} \Sigma_{inp} \text{ input actions} \\ \Sigma_{out} \text{ output actions} \end{array}\right\} \Sigma_{ext}$ externally observable

Σ_{int} *internal actions* cannot be observed

$\delta_a = \{(q, q') \mid (q, a, q') \in \delta\}$ set of a-*transitions*
(i.e. all arcs labeled with a)

Composable System

a set $\mathcal{S} = \{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$ of component automata is a *composable system* if $\forall\, i \in \{1, \ldots, n\}$:

$$\Sigma_{i,int} \cap \bigcup_{j \in \{1,\ldots,n\} \setminus \{i\}} \Sigma_j = \varnothing$$

$\Rightarrow$ internal actions are private (i.e. uniquely associated to one component automaton)

Fix: $\mathcal{S} = \{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$ is a composable system

? how to form a TA over a composable system

Complete Transition Space

The *complete transition space* of $a \in \Sigma = \bigcup_{i\in\{1,\ldots,n\}}(\Sigma_{i,inp} \cup \Sigma_{i,out} \cup \Sigma_{i,int})$ in $\mathcal{S}$ is

$$\Delta_a(\mathcal{S}) = \{(q,q') \in \prod_{i\in\{1,\ldots,n\}} Q_i \times \prod_{i\in\{1,\ldots,n\}} Q_i \mid$$

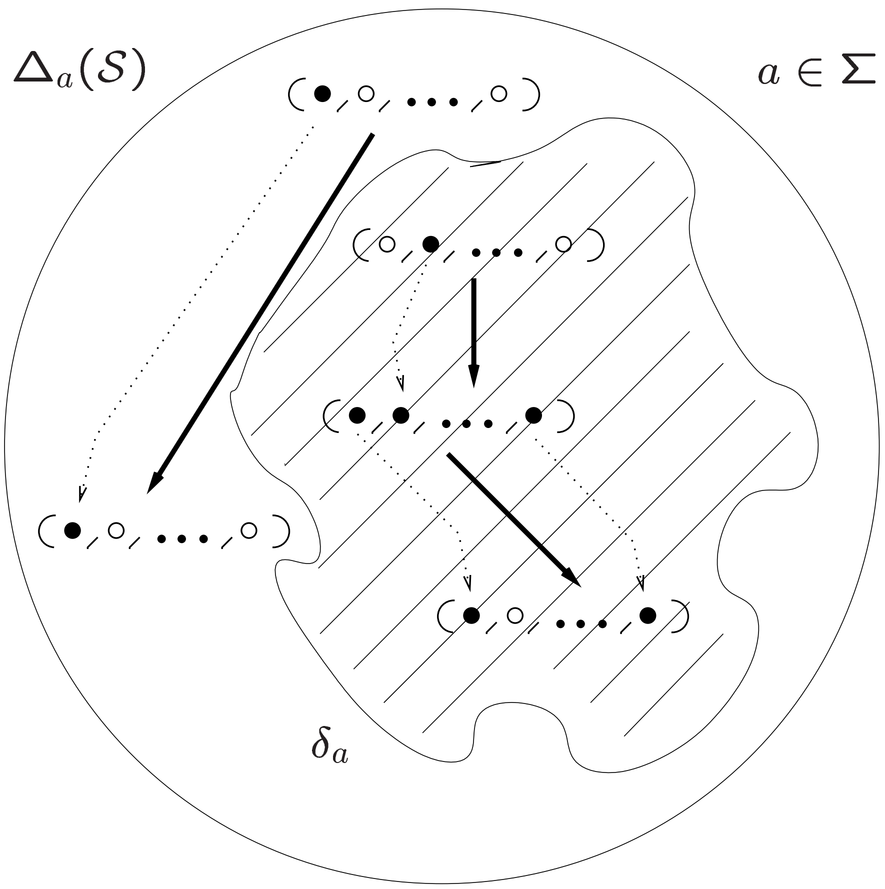
$$\exists j \in \{1,\ldots,n\} : \ (\mathbf{proj}_j(q), a, \mathbf{proj}_j(q')) \in \delta_j \wedge$$

$$\forall i \in \{1,\ldots,n\} : \ (\mathbf{proj}_i(q), a, \mathbf{proj}_i(q')) \in \delta_i \vee \mathbf{proj}_i(q) = \mathbf{proj}_i(q')\}$$

in every team transition <u>at least one</u> component acts <u>according</u> to its transition relation

all other components either <u>join</u> or are <u>idle</u>

Transition Space of TA



$\Rightarrow$ each <u>choice</u> of a team transition relation δ_a, $\forall a \in \Sigma$, defines a specific TA !

Team Automaton

$$\mathcal{T} = (\prod_{i\in\{1,\ldots,n\}} Q_i, (\Sigma_{inp}, \Sigma_{out}, \Sigma_{int}), \delta, \prod_{i\in\{1,\ldots,n\}} I_i)$$

is a *team automaton (TA) composed over* $\mathcal{S}$ if

$$\left.\begin{array}{l} \Sigma_{int} = \bigcup_{i\in\{1,\ldots,n\}} \Sigma_{i,int} \\[2ex] \Sigma_{out} = \bigcup_{i\in\{1,\ldots,n\}} \Sigma_{i,out} \\[2ex] \Sigma_{inp} = (\bigcup_{i\in\{1,\ldots,n\}} \Sigma_{i,inp}) \setminus \Sigma_{out} \end{array}\right\} = \Sigma$$

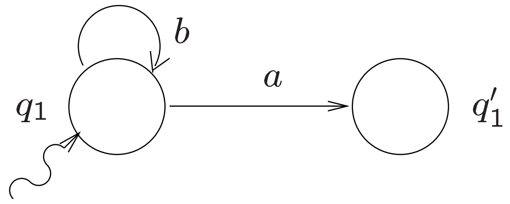
$\delta \subseteq \prod_{i\in\{1,\ldots,n\}} Q_i \times \Sigma \times \prod_{i\in\{1,\ldots,n\}} Q_i$ such that

$\forall a \in \Sigma$: $\quad \delta_a \subseteq \Delta_a(\mathcal{S})$
$\quad \& \ \ \delta_a = \Delta_a(\mathcal{S})$ if $a \in \Sigma_{int}$

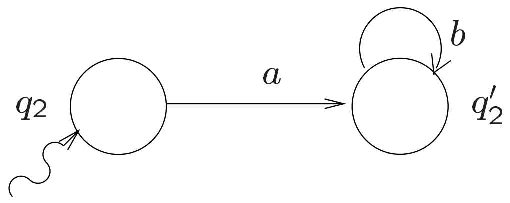
$\Rightarrow$ every team is a component & can thus be used to iteratively compose TA !

Example Component & Team Automata

$\mathcal{C}_1$:

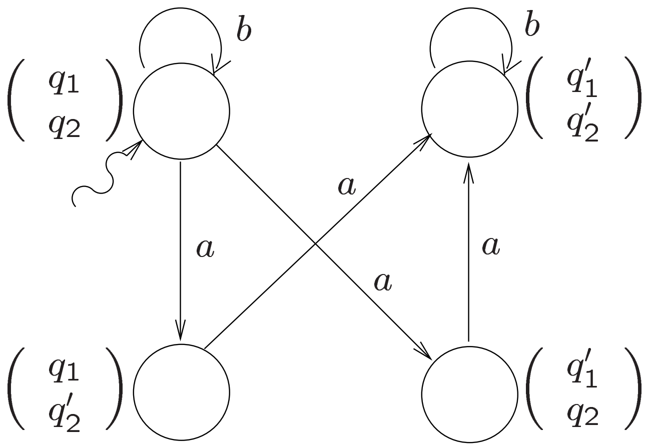


$\mathcal{C}_2$:

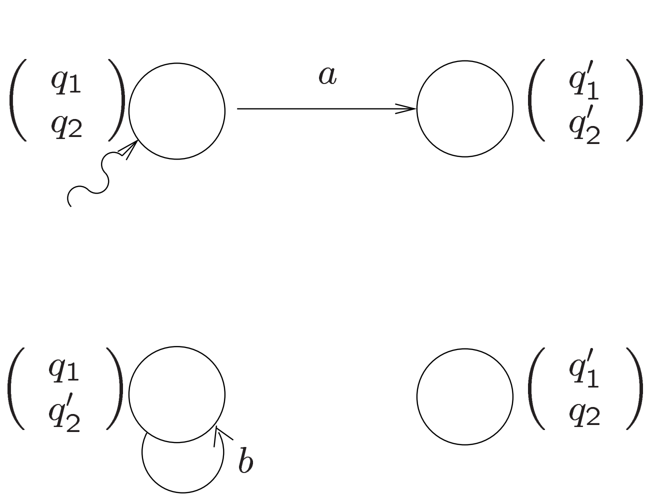


$\Rightarrow$ TA $\mathcal{T}^{free}$ & $\mathcal{T}^{ai}$ over the composable system $\{\mathcal{C}_1, \mathcal{C}_2\}$ defined by <u>choosing</u> their transitions!

$\mathcal{T}^{free}$:



$\mathcal{T}^{ai}$:

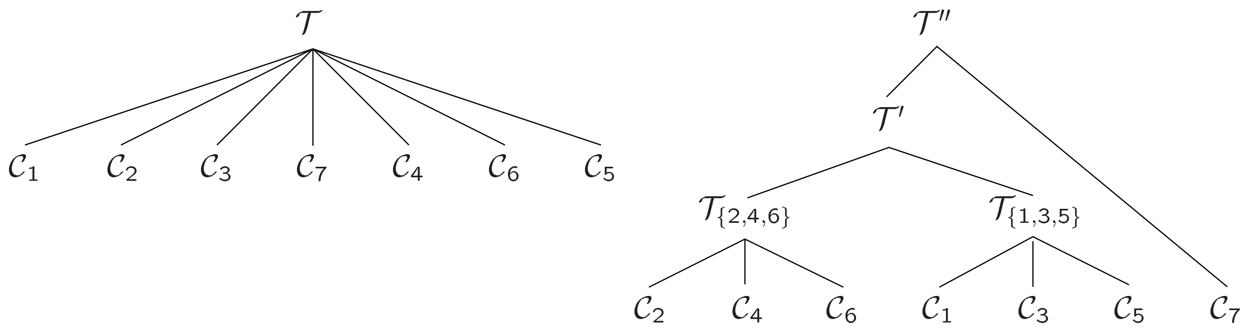


? who participates in $((q_1, q'_2), b, (q_1, q'_2))$ of $\mathcal{T}^{ai}$: only $\mathcal{C}_1$ or only $\mathcal{C}_2$ or both (<u>loop problem</u>)

$\Rightarrow$ adopt <u>maximal interpretation</u>: both execute b!

Teams over Teams

<u>Recall</u>: each TA can itself be used as a
component automaton



$\mathcal{T}$ is a TA over $\{\mathcal{C}_1, \ldots, \mathcal{C}_7\}$
$\mathcal{T}''$ is an *iterated* TA over $\{\mathcal{C}_1, \ldots, \mathcal{C}_7\}$

<u>Theorem</u>: after *reordering* elements of $\mathcal{T}''$:

$$\mathcal{T}'' \text{ is a TA over } \{\mathcal{C}_1, \ldots, \mathcal{C}_7\}$$

Key Notions so far

Component automaton

Composable system (of component automata)

Team automaton (over a composable system)

Subteam (of a team automaton)

Types of Synchronization

Basic: free, ai & si

Complex: peer-to-peer & master-slave

$\Rightarrow$ combinations thereof!

Fix TA by Synchronizations

In a TA $\mathcal{T}$ over $\mathcal{S}$ an action a is:

free if no a-transition of $\mathcal{T}$ contains the simultaneous execution of a by two or more components

ai (*action-indispensable*) if all components with a as an action participate in every a-transition of $\mathcal{T}$ (blocking can occur !)

$\mathcal{T}$ is the *max-free (ai) TA* over $\mathcal{S}$ if δ_a is the largest & hence unique transition relation in $\Delta_a(\mathcal{S})$ by which a is *free* (*ai*) !

$\Rightarrow$ maximal interpretation: $((q_1, q_2'), b, (q_1, q_2'))$ is contained in $\mathcal{T}^{ai}$, but not in $\mathcal{T}^{free}$!

Compositionality

"A specification of a composite system can be obtained from specifications of its components"

$\Rightarrow$ automata-based specification model: <u>behavior</u> !

[BK03] ter Beek & Kleijn

Team Automata Satisfying Compositionality

(FM 2003 - International Symposium of Formal Methods, 2003)

<u>Theorem</u>: if $\mathcal{T}$ is the *max-free* TA over $\mathcal{S}$ and $\mathcal{S}$ is loop limited, then

$$\mathbf{B}_{\mathcal{T}} = \|_{i\in\{1,\ldots,n\}} \mathbf{B}_{\mathcal{C}_i}$$

<u>Theorem</u>: if $\mathcal{T}$ is the *max-ai* TA over $\mathcal{S}$, then

$$\mathbf{B}_{\mathcal{T}} = \underline{\|}_{\{\Sigma_i \mid i\in\{1,\ldots,n\}\}} \mathbf{B}_{\mathcal{C}_i}$$

Indeed

$$\mathbf{B}_{\mathcal{T}^{ai}} = \{\lambda, a\} = \{b^n, b^n a \mid n \geq 0\} \;{}_{\Sigma_1}\underline{\|}_{\Sigma_2}\; \{\lambda, ab^n \mid n \geq 0\} = \mathbf{B}_{\mathcal{C}_1} \;{}_{\Sigma_1}\underline{\|}_{\Sigma_2}\; \mathbf{B}_{\mathcal{C}_2}$$

Related Models

Role of actions:

- input, output & internal: IOA, I/O systems, Interacting State Machines (ISMs), Reactive Transition Systems (RTSs) & CCS

Synchronizations:

- *uniform*: IOA, mixed product of automata, path expressions & COSY
- *pluriform*: Vector Controlled Concurrent Systems, free & synchronous product of automata
- other: CCS & (T)CSP

Composition:

- choice: synchronous product of automata
- *max-ai*-based: IOA, ISMs, RTSs, mixed product of automata, cooperating (pushdown) automata, timed cooperating automata, (T)CSP & statecharts

Applying TA

[BEKR03]

&

[HB00] 't Hoen & ter Beek

A Conflict-Free Strategy for Team-Based Model Development

(PDTSD'00 - International Workshop on Process support for Distributed Team-based Software Development, 2000)

&

[BEKR01b] ter Beek, Ellis, Kleijn & Rozenberg

Team Automata for Spatial Access Control

(ECSCW 2001 - European Conference on Computer-Supported Cooperative Work, 2001)

&

[BB03] ter Beek & Bloem

Model Checking Team Automata for Access Control

(unpublished manuscript, 2003)

&

Applying TA (continued)

[BLP03] ter Beek, Lenzini & Petrocchi

Team Automata for Security Analysis of Multicast/Broadcast Communication

(WISP'03 - Workshop on Issues in Security and Petri Nets, 2003)

&

[BCM03] ter Beek, Csuhaj-Varjú & Mitrana

Teams of Pushdown Automata

(PSI'03 - International Conference on Perspectives of System Informatics, 2003)

(International Journal of Computer Mathematics, 2004)

&

[BLP04] ter Beek, Lenzini & Petrocchi

Team Automata for Security Analysis

&

...