Games on Concurrent Processes: Epistemic Strategies

Konstantinos Chatzikokolakis
LIX, École Polytechnique

Sophia Knight
Prakash Panangaden
School of Computer Science, McGill University
Outline
Outline

• Introduction: Problems with traditional process algebra
Outline

• Introduction: Problems with traditional process algebra
• Games and Strategies
Outline

• Introduction: Problems with traditional process algebra

• Games and Strategies

• Epistemic restrictions on strategies
Outline

• Introduction: Problems with traditional process algebra
• Games and Strategies
• Epistemic restrictions on strategies
• Results
Outline

• Introduction: Problems with traditional process algebra
• Games and Strategies
• Epistemic restrictions on strategies
• Results
• Related work
Outline

• Introduction: Problems with traditional process algebra
• Games and Strategies
• Epistemic restrictions on strategies
• Results
• Related work
• Conclusion
Goals
Goals

• Use process algebra to model interacting agents in security protocols
Goals

• Use process algebra to model interacting agents in security protocols

• Control what the agents know by restricting their allowed strategies
Goals

• Use process algebra to model interacting agents in security protocols

• Control what the agents know by restricting their allowed strategies

• Understand and control information flow
Problems with Schedulers
Problems with Schedulers

• Traditionally, nondeterminism is resolved by a scheduler.
Problems with Schedulers

- Traditionally, nondeterminism is resolved by a scheduler.
- The scheduler is assumed to be omniscient.
Problems with Schedulers

• Traditionally, nondeterminism is resolved by a **scheduler**.

• The scheduler is assumed to be **omniscient**.

• It is hard to require it to respect **independence constraints** without controlling it somehow.
Problems with Schedulers
Problems with Schedulers

• A perverse scheduler can leak information to the outside world.
Problems with Schedulers

• A perverse scheduler can leak information to the outside world.

• Safety properties are required to hold with universal quantification over all possible schedulers,
Problems with Schedulers

• A perverse scheduler can leak information to the outside world.

• Safety properties are required to hold with universal quantification over all possible schedulers,

• So it is often impossible to prove certain security properties in this setting.
Example: Voting
Example: Voting
Example: Voting

- Two candidates: a, b
Example: Voting

- Two candidates: a, b
- Two voters: v, w
Example: Voting

- Two candidates: a, b
- Two voters: v, w
- Must output who voted, but not for whom they voted
Example: Voting

- Two candidates: a, b
- Two voters: v, w
- Must output who voted, but not for whom they voted
- Thick arrows show a scheduler that violates anonymity.
General Example
General Example

$$(\tau . P_1 + \tau . P_2) | (P_3 + P_4)$$
General Example

$$(\tau.P_1 + \tau.P_2) | (P_3 + P_4)$$

• Are the choices on the left and the right the same?
General Example

\((\tau.P_1 + \tau.P_2) \mid (P_3 + P_4)\)

- Are the choices on the left and the right the same?
- These choices might be made by different entities.
General Example

\((\tau.P_1 + \tau.P_2) \mid (P_3 + P_4)\)

• Are the choices on the left and the right the same?
• These choices might be made by different entities.
• Can one choice depend on the other?
Processes with Labels
Processes with Labels

\[ P, Q ::= 0 | l:a.P | P|Q | P + Q | (\nu a)P | l : \{P\} \]
What do Labels Mean?
What do Labels Mean?

• Labels represent what is visible about an action to the agent controlling the process.
What do Labels Mean?

• Labels represent what is visible about an action to the agent controlling the process.

• If two actions have the same label, they are indistinguishable to the agent, even if they are the same action.
What do Labels Mean?

- Labels represent what is visible about an action to the agent controlling the process.

- If two actions have the same label, they are indistinguishable to the agent, even if they are the same action.

- A process is deterministically labelled if there is never more than one action with the same label available.
What do Labels Mean?

- Labels represent what is visible about an action to the agent controlling the process.

- If two actions have the same label, they are indistinguishable to the agent, even if they are the same action.

- A process is deterministically labelled if there is never more than one action with the same label available.
Operational Semantics
Operational Semantics

Traditional CCS Semantics,

plus the SWITCH rule
The SWITCH Rule

\[
\frac{P \xrightarrow{\tau} P'}{l: \{P\} \xrightarrow{\tau} P'}
\]
The SWITCH Rule

\[
P \xrightarrow{\tau} P'
\]

\[
l : \{P\} \xrightarrow{\tau} P'
\]

- Represents choices made \textit{independently} from other choices in the process.
The SWITCH Rule

\[ P \xrightarrow{\tau} P' \]
\[ l: \{ P \} \xrightarrow{\tau} P' \]

- Represents choices made independently from other choices in the process.

- There are two agents making choices. The \{\} operator represents control switching from one to the other for one step.
Protection Example
Protection Example

\[ \{ \tau P_1 + \tau P_2 \} \mid (P_3 + P_4) \]
Protection Example

\{\tau.P_1 + \tau.P_2\} | (P_3 + P_4)

• This means that the two choices are independent.
Protection Example

\[ \{ \tau.P_1 + \tau.P_2 \} \mid (P_3 + P_4) \]

- This means that the two choices are independent.
- They may be controlled by different entities.
Protection Example

\{ \tau . P_1 + \tau . P_2 \} \mid (P_3 + P_4) 

• This means that the two choices are independent.

• They may be controlled by different entities.

• Neither choice can directly depend on the outcome of the other.
Games
Games

• One game is defined for each specific process: the process is the game board.
Games

• One game is defined for each specific process: the process is the game board.

• No concept of winning or losing.
Games

• One game is defined for each specific process: the process is the game board.

• No concept of winning or losing.

• Two players: $X$ and $Y$. Two players are sufficient to model independence and interaction.
Games

• One game is defined for each specific process: the process is the game board.

• No concept of winning or losing.

• Two players: $X$ and $Y$. Two players are sufficient to model independence and interaction.

• Players are independent and act according to their strategies.
Games

• One game is defined for each specific process: the process is the game board.

• No concept of winning or losing.

• Two players: $X$ and $Y$. Two players are sufficient to model independence and interaction.

• Players are independent and act according to their strategies.

• Players’ interaction determines how the process executes.
Moves and Valid Positions
Moves and Valid Positions

• Players’ moves in the game are process labels.
Moves and Valid Positions

- Players’ moves in the game are process labels.
- Each move belongs to one player.
Moves and Valid Positions

• Players’ moves in the game are process labels.
• Each move belongs to one player.
• A string of allowable moves is called a valid position.
Moves and Valid Positions

• Players’ moves in the game are process labels.

• Each move belongs to one player.

• A string of allowable moves is called a valid position.

• A valid position is like a trace, but with labels instead of actions.
Valid Positions: Example

\[ P = 1 \{ 2\tau \cdot 3a + 4\tau \cdot 3b \} | (5a + 6b) \]

Example valid positions:

1.2.3.5, 1.2.3.6, 1.4.3.5, 1.4.3.6, 1.2.5.3, ..., 5.1.2.3, ...
Valid Positions: Example

\[ P = \frac{1}{2} \left\{ 2\tau \cdot 3a + 4\tau \cdot 3b \right\} |\left( 5a + 6b \right) \]

Example valid positions:

1.2.3.5, 1.2.3.6, 1.4.3.5, 1.4.3.6, 1.2.5.3, ..., 5.1.2.3, ...

• Prefix closed.
Valid Positions: Example

\[ P = \{ \tau \cdot a + \tau \cdot b \} | (a + b) \]

Example valid positions:

1.2.3.5, 1.2.3.6, 1.4.3.5, 1.4.3.6, 1.2.5.3, ..., 5.1.2.3, ...

• Prefix closed.

• Every valid position represents a unique execution.
Strategies
Strategies

• Belongs to one player, Z.
Strategies

- Belongs to one player, Z.
- Tells Z what move to make in certain executions of the process.
Strategies

• Belongs to one player, $Z$.

• Tells $Z$ what move to make in certain executions of the process.

• A set of valid positions, each ending with a move for $Z$. 
Strategies

• Belongs to one player, Z.

• Tells Z what move to make in certain executions of the process.

• A set of valid positions, each ending with a move for Z.

• “Prefix closed,” but only for Z’s own valid positions.
Strategy Examples

\[ P = \frac{1}{1} \{ 2\tau \cdot 3a + 4\tau \cdot 3b \} \mid (5a + 6b) \]

A strategy for \( X \): \( \{1, 1.2.3, 1.2.3.5\} \)

A strategy for \( Y \): \( \{1.2, 1.4, 5.1.2, 6.1.4\} \)
Restriction: Determinacy
Restriction: Determinacy

- Strategy must not tell the player to make more than one move at once.
Restriction: Determinacy

- Strategy must not tell the player to make more than one move at once.

\[
\begin{align*}
  s.m_1 & \in S \\
  s.m_2 & \in S \\
\end{align*}
\] \Rightarrow m_1 = m_2
Restriction: Completeness
Restriction: Completeness

• Strategy must tell player what to do in every possible situation.
Restriction: Completeness

• Strategy must tell player what to do in every possible situation.

• Definition: if a position is reachable by following the strategy, then the strategy must tell the player what to do in that position.
Executions
Executions

• If each player follows a deterministic, complete strategy, it defines a unique valid position, representing a unique execution of the process.
Example Execution

\[ P = \frac{1}{1} \{ \tau \cdot 3a + \tau \cdot 3b \} \mid (a + b) \]
Example Execution

\[ P = 1 \{ 2 \tau \cdot 3a + 4 \tau \cdot 3b \} | (5a + 6b) \]

A strategy for X: 1.2.3.5, 1.4.3.6,
(plus appropriate prefixes).
Example Execution

\[ P = \frac{1}{2} \{ 2\tau \cdot 3a + 4\tau \cdot 3b \} | (5a + 6b) \]

A strategy for X: 1.2.3.5, 1.4.3.6, (plus appropriate prefixes).

A strategy for Y: 1.2, 5.1.4, 6.1.4.
Example Execution

\[ P = 1\{2\tau \cdot 3a + 4\tau \cdot 3b\} \mid (5a + 6b) \]

A strategy for X: 1.2.3.5, 1.4.3.6, (plus appropriate prefixes).

A strategy for Y: 1.2, 5.1.4, 6.1.4.

The execution determined by these strategies is \( \tau.a.a \), from the valid position 1.2.3.5.
Epistemic Restrictions
Epistemic Restrictions

• Strategies can only be based on what players know.
Epistemic Restrictions

• Strategies can only be based on what players know.

• For each player, we define an equivalence on valid positions, capturing the player’s knowledge.
Epistemic Restrictions

• Strategies can only be based on what players know.

• For each player, we define an equivalence on valid positions, capturing the player’s knowledge.

• Add this restriction:
Epistemic Restrictions

• Strategies can only be based on what players know.

• For each player, we define an equivalence on valid positions, capturing the player’s knowledge.

• Add this restriction:

\[ s_1 \sim s_2 \text{ then } s_1.m \in S \iff s_2.m \in S. \]
Epistemic Restrictions

• Strategies can only be based on what players know.

• For each player, we define an equivalence on valid positions, capturing the player’s knowledge.

• Add this restriction:

\[ s_1 \sim s_2 \text{ then } s_1.m \in S \iff s_2.m \in S. \]
Epistemic Restrictions

• Strategies can only be based on what players know.

• For each player, we define an equivalence on valid positions, capturing the player’s knowledge.

• Add this restriction:

\[ s_1 \sim s_2 \text{ then } s_1.m \in S \iff s_2.m \in S. \]

• One can design different equivalences to “engineer” the appropriate epistemic concept.
Introspection
Introspection

• An example epistemic restriction: introspection.
Introspection

• An example epistemic restriction: introspection.

• The player knows his own history and what moves were available to him at every point in the past.
The Main Technical Result
The Main Technical Result

- The introspective restriction exactly captures the independence requirement that one expects
The Main Technical Result

- The introspective restriction exactly captures the independence requirement that one expects.
- In particular, they are equivalent to the syntactic schedulers of Chatzikokolakis and Palamidessi.
Related Work
Related Work

- Epistemic logic much used in the distributed systems community, we have used the same semantics but we have not incorporated the formal logic. No tie up with a process algebra or any compositional formalism in the distributed systems community.
Related Work

• Epistemic logic much used in the distributed systems community, we have used the same semantics but we have not incorporated the formal logic. No tie up with a process algebra or any compositional formalism in the distributed systems community.

• In a recent paper, Chadha, Delaune and Kremer developed an epistemic logic to talk about executions in the applied Pi-calculus.
• Deschesne, Mousavi and Orzan (LPAR 07) have a rich formalism combining epistemic logic and process algebra.
• Deschesne, Mousavi and Orzan (LPAR 07) have a rich formalism combining epistemic logic and process algebra.

• Hyland and Ong have a discussion of representing strategies as terms in the pi-calculus. They do not have a semantics of the process language itself.
• Deschesne, Mousavi and Orzan (LPAR 07) have a rich formalism combining epistemic logic and process algebra.

• Hyland and Ong have a discussion of representing strategies as terms in the pi-calculus. They do not have a semantics of the process language itself.

• Game semantics models programs as strategies; in our work the programs (processes) are the game board.
Future Work
Future Work

- Extension to recursion and mobile calculi.
Future Work

• Extension to recursion and mobile calculi.

• Formal epistemic, temporal logic for processes, with connections to bisimulation.
Future Work

• Extension to recursion and mobile calculi.

• Formal epistemic, temporal logic for processes, with connections to bisimulation.

• Extension to probabilistic process algebra (already worked out).
Future Work

- Extension to recursion and mobile calculi.
- Formal epistemic, temporal logic for processes, with connections to bisimulation.
- Extension to probabilistic process algebra (already worked out).
- Combining probabilistic epistemic reasoning and information theory.
Conclusions
Conclusions

- A semantic description of limitations of the power of agents.
Conclusions

• A semantic description of limitations of the power of agents.

• A framework that can be used for other such limitations.
Conclusions

• A semantic description of limitations of the power of agents.

• A framework that can be used for other such limitations.

• A better conceptual understanding of knowledge and interaction.