Quantifying Information Flow
Gavin Lowe

Nate Nystrom
CS 711
November 24, 2003
Introduction

- Some information flow is inevitable and acceptable.
- Previous work: can “low” user distinguish between two different behaviors of a “high” user to pass at least one bit of information?
- This work: how much information flows from “high” to “low”?
- Uses a process algebra approach (Timed CSP) to define the information flow quantity
Outline

• Timed CSP
• Examples
• Information flow quantity
• No information flow
• Bounded-time information flow
• Two users: High and Low.
• High is malicious: he wants to pass information to Low.
• *Information flow quantity* (IFQ) is the number of behaviors of High that are distinguishable from Low’s point of view.
• If there are $N$ such behaviors, then High can use the system to pass $\log_2 N$ bits of information to Low.
• Note: $\log_2 1 = 0$, so an absence of information flow is represented by an IFQ of 1.
Timed CSP

• A process $P$ offers to participate in events, or may refuse events

• Events represent atomic communication between two processes
  ▶ High events $h \in H$; low events $l \in L$; $H \cap L = \emptyset$
  ▶ $\Sigma = H \cup L$ is the set of standard events
  ▶ tock event represents passage of one time step
    * All processes participate in tock; none can refuse it
  ▶ $\Sigma_{tock} = \Sigma \cup \{tock\}$

• Channels $c$ carry sets of events
  ▶ $c.5$ is an event of channel $c$
## Timed CSP

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STOP</td>
<td>perform no events</td>
</tr>
<tr>
<td>\text{WAIT } t; \ P</td>
<td>do nothing for $t$ time steps, then act like $P$</td>
</tr>
<tr>
<td>$a \rightarrow P$</td>
<td>perform event $a$, then act like $P$</td>
</tr>
<tr>
<td>$P \Box Q$</td>
<td>external ND choice decided by environment</td>
</tr>
<tr>
<td>$P \cap Q$</td>
<td>internal ND choice outside the model</td>
</tr>
<tr>
<td>$P \triangleright Q$</td>
<td>act like $P$, become $Q$ after $t$ if no event occurs</td>
</tr>
<tr>
<td>$P \setminus A$</td>
<td>act like $P$, but hide events of $A$</td>
</tr>
<tr>
<td>\text{RUN}(A)</td>
<td>perform any events of $A$, but never refuse $A$</td>
</tr>
<tr>
<td>\text{CHAOS}(A)</td>
<td>perform any events of $A$, and refuse any</td>
</tr>
<tr>
<td>$P</td>
<td></td>
</tr>
</tbody>
</table>
Example

\[ P_1 \equiv h \rightarrow l \rightarrow \text{STOP} \rightarrow \text{STOP} \]

- Perform \( h \), then perform \( l \), then stop.
- Or: if \( h \) not performed in one step, stop.
- \( IFQ(P_1) = 2 \)
  - If High performs \( h \) within the first time step, then Low can perform an \( l \).
  - If High does not perform an \( h \) within the first time step, then Low will see that the event was refused up to the first \( \text{tock} \).

\[ \Rightarrow \text{High can use } P_1 \text{ to pass one bit of information to Low.} \]
Example: timing channels

\[ P_2 \triangleq h \rightarrow l \rightarrow \text{STOP} \overset{N}{\triangleright} \text{STOP} \]

- Perform \( h \), then perform \( l \), then stop.
- Or: if \( h \) not performed in \( N \) steps, stop.
- \( \text{IFQ}(P_1) = N + 1 \)
  - High can pass a value \( k \in \{0, \ldots, N - 1\} \) by performing \( h \) at time \( k \). Low will observe \( k \) tocks, then can perform \( l \).
  - High can pass an additional value by not performing any event in the first \( N \) steps.
- In \( P_1 \), Low can only tell \textit{whether} High performed an event.
- In \( P_2 \), Low can tell \textit{when} High performed an event.
Nondeterminism

• Previous work modeled nondeterminism probabilistically
• Better: consider all possible ways nondeterminism can be resolved, and use the *worst case*
• Two types of nondeterminism:
  ▶ “don’t care”: $P \Box Q$ is an *external* choice resolved by the environment when the initial event of $P$ or $Q$ is performed:

  \[
  \frac{P_1 \rightarrow P'_1}{P_1 \Box P_2 \rightarrow P'_1 \Box P_2} \quad \frac{P_2 \rightarrow P'_2}{P_1 \Box P_2 \rightarrow P'_1 \Box P'_2} \quad \frac{P_1 \overset{a}{\rightarrow} P'_1}{P_1 \Box P_2 \overset{a}{\rightarrow} P'_1} \quad \frac{P_2 \overset{a}{\rightarrow} P'_2}{P_1 \Box P_2 \overset{a}{\rightarrow} P'_2}
  \]

  ▶ “don’t know”: $P \ominus Q$ is an *internal* choice resolved silently by something outside the model:

  \[
  \frac{P_1 \ominus P_2 \rightarrow P_1}{P_1 \ominus P_2 \rightarrow P_1} \quad \frac{P_1 \ominus P_2 \rightarrow P_2}{P_1 \ominus P_2 \rightarrow P_2}
  \]
Example: nondeterminism

\[ P_3 = \left( \left( h_1 \rightarrow (l_1 \rightarrow \text{STOP} \sqcap l_2 \rightarrow \text{STOP}) \sqcap h_2 \rightarrow (l_1 \rightarrow \text{STOP} \sqcap l_2 \rightarrow \text{STOP}) \right) \right)^1 \triangleright \text{STOP} \]

- \( IFQ(P_3) = 3 \)
  - Low can tell whether or not High has performed some event.
  - Can Low distinguish the two behaviors of the system following \( h_1 \) and \( h_2 \)?
  - Best case: If the two nondeterministic choices (\( \sqcap \)) were implemented identically, then \( IFQ(P_3) = 2 \)
  - Worst case: If the first choice always selects the first argument and the second always selects the second, then \( IFQ(P_3) = 3 \).
Refusal traces

• A refusal is either
  ▶ a set $X$ of events, meaning events of $X$ are unavailable, or
  ▶ the null refusal, $\bullet$, meaning nothing is refused

• A refusal trace is an alternating sequence of refusals and events:

$$\{b\} \xrightarrow{a} \bullet \xrightarrow{tack} \{a, b\}$$

refuses $b$, performs $a$, refuses nothing, performs tack, refuses $a$ and $b$.

• $\mathcal{R}[P]$ is the set of refusal traces of $P$
Low’s strategy

• Low interacts with the system $S$ through a test process $T$, which repeatedly offers events in $L \cup \{tock\}$

• $S$ and $T$ are composed like this: $(S \parallel_L T) \setminus L$

• $T$ gives results on channel $\omega$ via events $\omega.k \not\in \Sigma$

\[
\text{results}(S, T) \overset{\Delta}{=} \{ k : \exists n \in \mathbb{N}. \bullet (\xrightarrow{tock} \bullet)^n \xrightarrow{\omega.k} \bullet \in \mathcal{R}[(S \parallel_L T) \setminus L] \}
\]

i.e., $k$ such that the refusal trace of the composition starts with an arbitrary number of $tock$ events, then the event $\omega.k$ is performed.
High’s strategy

- Model High’s behavior by a process $Q$ with alphabet $\Sigma_{tock}$.
- High’s behavior includes the behavior of the scheduler.
- Low’s view of the system is given by $(P ||_{\Sigma} Q) \setminus H$.

Example: $P_5 = (l \rightarrow \text{STOP} \; \square h \rightarrow \text{STOP}) \; 1 \succ \text{STOP}$

- High could pass one value by performing $h$: $Q \wedge h \rightarrow \text{STOP}$
- Or, High can pass a different value by not performing $h$: $Q \wedge l \rightarrow \text{STOP}$
Combining the strategies

• To pass value $k$ to Low, High will act like process $Q(k)$
• Low’s possible views of the system are the set:
  \[
  \{(P \parallel \Sigma Q(k)) \setminus H : k \in \text{dom}(Q)\}
  \]
• Can a particular test $T$ for Low distinguish these processes?
• Define
  \[
  \text{results}(P, Q, T) \triangleq \text{results}((P \parallel \Sigma Q) \setminus H, T)
  \]
• Should only consider strategies where if High wants to send $k$, then Low gets results $k$; that is,
  \[
  \text{ok}(P, Q, T) \triangleq \forall k \in \text{dom}(Q). \text{results}(P, Q(k), T) = \{k\}
  \]
  (and some other conditions I’m leaving out)
Example

• Consider the process:

\[ P_1 \equiv h \rightarrow l \rightarrow \text{STOP} \uparrow^1 \text{STOP} \]

• and the strategy:

\[ Q(0) \equiv \text{RUN}(L) \]
\[ Q(1) \equiv h \rightarrow \text{RUN}(L) \]
\[ T \equiv l \rightarrow \text{SUCCESS}(1) \uparrow^1 \text{SUCCESS}(0) \]

where \( \text{SUCCESS}(k) \equiv \omega.k \rightarrow \text{STOP} \)

• \( (P_1 \parallel \Sigma Q(0)) \setminus H \) behaves like \( \text{STOP} \)
  \[ \Rightarrow \text{results}(P_1, Q(0), T) = \{0\} \]

• \( (P_1 \parallel \Sigma Q(1)) \setminus H \) behaves like \( l \rightarrow \text{STOP} \)
  \[ \Rightarrow \text{results}(P_1, Q(1), T) = \{1\} \]
Defining IFQ

- Given some process $P$ and some strategy $Q$ and test $T$, such that $\text{ok}(P, Q, T)$, the associated flow is the number of different values that can be sent, i.e., $\# \text{dom}(Q)$.

- But, want to consider, not just $P$, but all refinements $R$ of $P$ to account for possible ways nondeterminism is resolved:

\[
P \equiv_T P' \triangleq \forall T. \text{results}(P, T) = \text{results}(P', T)
\]

\[
P \sqsubseteq_T R \triangleq \forall T. \text{results}(P, T) \supseteq \text{results}(R, T)
\]

- Then, assume the worst case scenario:

\[
\text{IFQ}(P) \triangleq \max\{\# \text{dom}(Q) : P \sqsubseteq_T R \land \text{ok}(R, Q, T)\}
\]
Example

- Consider the process:

\[ P_5 \equiv (l \rightarrow \text{STOP} \Box h \rightarrow \text{STOP}) \uparrow \text{STOP} \]

- and the strategy:

\[ Q(0) \equiv \text{RUN}(L) \]
\[ Q(1) \equiv h \rightarrow \text{RUN}(L) \]

\[ T \equiv l \rightarrow \text{SUCCESS}(0) \uparrow \text{SUCCESS}(1) \]

- Note:

\[ \text{results}(P_5, Q(0), T) = \{0\} \quad \text{and} \quad \text{results}(P_5, Q(1), T) = \{1\} \]

- \( \text{IFQ}(P_5) = \# \text{dom}(Q) = 2 \).
When is $IFQ = 1$?

- $P$ satisfies testing nondeducibility on composition (TNDC) iff:
  \[
  \forall Q \in CSP_H. P \parallel_H STOP \equiv_T (P \parallel_H Q) \setminus H
  \]
where $CSP_H$ is the set of processes with alphabet $H \cup \{tock\}$.

- They strengthen TNDC to strong testing nondeducibility on composition (STNDC). $P$ satisfies STNDC iff:
  \[
  \forall R \sqsubseteq_T P. R \text{ satisfies TDNC}
  \]

- Let $\text{LEAK}$ be an insecure process. $\text{LEAK} \sqcap \text{CHAOS}(L)$ satisfies TNDC, but not STNDC.
  - This program is analogous to: $l := h \Box l := \text{rand}(2)$

- Main result: their definition of IFQ gives IFQ of 1 to precisely those processes that satisfy STNDC.
Bounded-time information flow

- Given time, some process may pass unbounded information
- Want to compute the rate of information flow
- Define results obtainable from $S$ with test $T$ before time $t+1$:
  \[ \text{results}_t(S, T) \triangleq \{ k : \exists n \leq t. \bullet (\xrightarrow{\text{tock}} \bullet)^n \xrightarrow{\omega, k} \bullet \in \mathcal{R}(S ||_L T) \setminus L \} \]
- Can analogously define:
  \[ \text{results}_t(P, Q, T) \triangleq \text{results}_t(((P || \Sigma Q) \setminus H, T) \]
  \[ \text{ok}_t(P, Q, T) \triangleq \forall k \in \text{dom}(Q). \text{results}_t(P, Q(k), T) = \{ k \} \]
  \[ \text{IFQ}_t(P) \triangleq \max\{ \# \text{dom}(Q) : P \sqsubseteq_T R \land \text{ok}_t(R, Q, T) \} \]
- Then, long term information flow rate is:
  \[ LTIFR(P) \triangleq \lim_{t \to \infty} \frac{\log_2 \text{IFQ}_t(P)}{t} \text{ bits per step} \]
Example

• Consider:

\[ P \triangleq h \rightarrow l \rightarrow \text{STOP} \]

• Fix \( N \) and consider the strategy:

\[ Q(k) = \text{WAIT } k; h \rightarrow l \rightarrow \text{STOP} \quad \text{for } k = 0, \ldots, N - 1 \]
\[ Q(N) = \text{STOP} \]
\[ T(k) = l \rightarrow \text{SUCCESS}(k) \uparrow T(k + 1) \quad \text{for } k = 0, \ldots, N - 1 \]
\[ T(N) = \text{SUCCESS}(N) \]

• Low cannot distinguish more than \( N \) behaviors in \( N \) tocks.

• Therefore \( IFQ_N(P) = N + 1 \), and \( LTIFR(P) = 0 \).
Conclusions

- Defined information flow quantity (IFQ) for a process $P$ to be the number of behaviors of High observable by Low.
- Defined a criterion (strong testing nondeducibility on composition) for which IFQ is 1.
- Defined information flow rate.