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

Information Flow Quantity, informally

- 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

$\mathtt{STOP}$	perform no events
$\mathtt{WAIT}\ t;\ P$	do nothing for t time steps, then act like P
$a \longrightarrow P$	perform event a, then act like P
$P \,\Box\, Q$	*external* ND choice decided by environment
$P \sqcap Q$	*internal* ND choice outside the model
$P \overset{t}{\triangleright} Q$	act like P, become Q after t if no event occurs
$P \setminus A$	act like P, but hide events of A
$\mathtt{RUN}(A)$	perform any events of A, but never refuse A
$\mathtt{CHAOS}(A)$	perform any events of A, and refuse any
$P \mid\mid_A Q$	run P and Q in parallel, sync on $A \cup \{tock\}$

Example

$$P_1 \mathrel{\widehat{=}} h \rightarrow l \rightarrow \mathtt{STOP} \overset{1}{\triangleright} \mathtt{STOP}$$

- 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 $tock$.

$\Rightarrow$ High can use P_1 to pass one bit of information to Low.

Example: timing channels

$$P_2 \mathrel{\widehat{=}} h \to l \to \mathtt{STOP} \overset{N}{\rhd} \mathtt{STOP}$$

- Perform h, then perform l, then stop.
- Or: if h not performed in N steps, stop.
- $IFQ(P_1) = N + 1$
  - 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 *whether* High performed an event.
- In P_2, Low can tell *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:

$$\frac{P_1 \xrightarrow{\tau} P_1'}{P_1 \,\Box\, P_2 \xrightarrow{\tau} P_1' \,\Box\, P_2} \qquad \frac{P_2 \xrightarrow{\tau} P_2'}{P_1 \,\Box\, P_2 \xrightarrow{\tau} P_1 \,\Box\, P_2'} \qquad \frac{P_1 \xrightarrow{a} P_1'}{P_1 \,\Box\, P_2 \xrightarrow{a} P_1'} \qquad \frac{P_2 \xrightarrow{a} P_2'}{P_1 \,\Box\, P_2 \xrightarrow{a} P_2'}$$

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

$$\frac{}{P_1 \sqcap P_2 \xrightarrow{\tau} P_1} \qquad \frac{}{P_1 \sqcap P_2 \xrightarrow{\tau} P_2}$$

Example: nondeterminism

$$P_3 = \left(\begin{array}{l} h_1 \longrightarrow (l_1 \longrightarrow \mathtt{STOP} \sqcap l_2 \longrightarrow \mathtt{STOP} \\ \;\square\; h_2 \longrightarrow (l_1 \longrightarrow \mathtt{STOP} \sqcap l_2 \longrightarrow \mathtt{STOP}) \end{array} \right) \overset{1}{\triangleright} \mathtt{STOP}$$

- $IFQ(P_3) = 3$
  - 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:

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

  refuses b, performs a, refuses nothing, performs $tock$, 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: $(S \mathbin{||_L} T) \setminus L$
- T gives results on channel ω via events $\omega.k \notin \Sigma$

$$results(S,T) \mathrel{\widehat{=}} \{k : \exists n \in \mathbb{N}.\, \bullet\, (\xrightarrow{tock} \bullet)^n \xrightarrow{\omega.k} \bullet \in \mathcal{R}[\![(S \mathbin{||_L} T) \setminus L]\!]\}$$

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 Σ_{tock}.
- High's behavior includes the behavior of the scheduler.
- Low's view of the system is given by $(P \mid\mid_{\Sigma} Q) \setminus H$.
- Example: $P_5 = (l \longrightarrow \mathtt{STOP} \,\Box\, h \longrightarrow \mathtt{STOP}) \overset{1}{\triangleright} \mathtt{STOP}$
  - High could pass one value by performing h:

$$Q \mathrel{\widehat{=}} h \longrightarrow \mathtt{STOP}$$

  - Or, High can pass a different value by not performing h:

$$Q \mathrel{\widehat{=}} l \longrightarrow \mathtt{STOP}$$

Combining the strategies

- To pass value k to Low, High will act like process $\mathcal{Q}(k)$
- Low's possible views of the system are the set:

$$\{(P \mid\mid_{\Sigma} \mathcal{Q}(k)) \setminus H : k \in \mathrm{dom}(\mathcal{Q})\}$$

- Can a particular test T for Low distinguish these processes?
- Define

$$\mathit{results}(P, Q, T) \mathrel{\widehat{=}} \mathit{results}((P \mid\mid_{\Sigma} Q) \setminus H, T)$$

- Should only consider strategies where if High wants to send k, then Low gets results k; that is,

$$ok(P, \mathcal{Q}, T) \mathrel{\widehat{=}} \forall k \in \mathrm{dom}(\mathcal{Q}) . \mathit{results}(P, \mathcal{Q}(k), T) = \{k\}$$

(and some other conditions I'm leaving out)

Example

- Consider the process:

$$P_1 \mathrel{\widehat{=}} h \longrightarrow l \longrightarrow \mathtt{STOP} \overset{1}{\triangleright} \mathtt{STOP}$$

- and the strategy:

$$\mathcal{Q}(0) \mathrel{\widehat{=}} \mathtt{RUN}(L)$$
$$\mathcal{Q}(1) \mathrel{\widehat{=}} h \longrightarrow \mathtt{RUN}(L)$$

$$T \mathrel{\widehat{=}} l \longrightarrow \mathtt{SUCCESS}(1) \overset{1}{\triangleright} \mathtt{SUCCESS}(0)$$

where $\mathtt{SUCCESS}(k) \mathrel{\widehat{=}} \omega.k \longrightarrow \mathtt{STOP}$

- $(P_1 \,||_\Sigma\, \mathcal{Q}(0)) \setminus H$ behaves like $\mathtt{STOP}$
  $\Rightarrow results(P_1, \mathcal{Q}(0), T) = \{0\}$
- $(P_1 \,||_\Sigma\, \mathcal{Q}(1)) \setminus H$ behaves like $l \longrightarrow \mathtt{STOP}$
  $\Rightarrow results(P_1, \mathcal{Q}(1), T) = \{1\}$

Defining IFQ

- Given some process P and some strategy $\mathcal{Q}$ and test T, such that $ok(P, \mathcal{Q}, T)$, the associated flow is the number of different values that can be sent, i.e., $\#\,\mathrm{dom}(\mathcal{Q})$.
- But, want to consider, not just P, but all *refinements* R of P to account for possible ways nondeterminism is resolved:

$$P \equiv_T P' \mathrel{\widehat{=}} \forall T.results(P, T) = results(P', T)$$
$$P \sqsubseteq_T R \mathrel{\widehat{=}} \forall T.results(P, T) \supseteq results(R, T)$$

- Then, assume the worst case scenario:

$$IFQ(P) \mathrel{\widehat{=}} \max\{\#\,\mathrm{dom}(\mathcal{Q}) : P \sqsubseteq_T R \wedge ok(R, \mathcal{Q}, T)\}$$

Example

- Consider the process:

$$P_5 \mathrel{\widehat{=}} (l \longrightarrow \mathtt{STOP} \mathbin{\Box} h \longrightarrow \mathtt{STOP}) \overset{1}{\rhd} \mathtt{STOP}$$

- and the strategy:

$$\mathcal{Q}(0) \mathrel{\widehat{=}} \mathtt{RUN}(L)$$
$$\mathcal{Q}(1) \mathrel{\widehat{=}} h \longrightarrow \mathtt{RUN}(L)$$

$$T \mathrel{\widehat{=}} l \longrightarrow \mathtt{SUCCESS}(0) \overset{1}{\rhd} \mathtt{SUCCESS}(1)$$

- Note:

$$results(P_5, \mathcal{Q}(0), T) = \{0\} \text{ and } results(P_5, \mathcal{Q}(1), T) = \{1\}$$

- $\mathit{IFQ}(P_5) = \#\,\mathrm{dom}(\mathcal{Q}) = 2$.

When is $IFQ = 1$?

- P satisfies *testing nondeducibility on composition (TNDC)* iff:

$$\forall Q \in CSP_H . P \mid\mid_H \mathtt{STOP} \equiv_T (P \mid\mid_H Q) \setminus H$$

  where CSP_H is the set of processes with alphabet $H \cup \{tock\}$.

- They strengthen TNDC to *strong testing nondeducability on composition (STNDC)*. P satisfies STNDC iff:

$$\forall R \sqsupseteq_T P . R \text{ satisfies TDNC}$$

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

- 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 $t+1$:

$$results_t(S,T) \mathrel{\widehat{=}} \{k : \exists n \leq t.\, \bullet (\xrightarrow{tock} \bullet)^n \xrightarrow{\omega.k} \bullet \in \mathcal{R}[\![(S \,||_L\, T) \setminus L]\!]\}$$

- Can analogously define:

$$\begin{aligned} results_t(P,\mathcal{Q},T) &\mathrel{\widehat{=}} results_t((P \,||_\Sigma\, Q) \setminus H, T) \\ ok_t(P,\mathcal{Q},T) &\mathrel{\widehat{=}} \forall k \in \mathrm{dom}(\mathcal{Q}).\, results_t(P,\mathcal{Q}(k),T) = \{k\} \\ IFQ_t(P) &\mathrel{\widehat{=}} \max\{\#\,\mathrm{dom}(\mathcal{Q}) : P \sqsubseteq_T R \wedge ok_t(R,\mathcal{Q},T)\} \end{aligned}$$

- Then, *long term information flow rate* is:

$$LTIFR(P) \mathrel{\widehat{=}} \lim_{t \to \infty} \frac{\log_2 IFQ_t(P)}{t} \text{ bits per step}$$

Example

- Consider:

$$P \mathrel{\widehat{=}} h \rightarrow l \rightarrow \texttt{STOP}$$

- Fix N and consider the strategy:

$$\begin{aligned}\mathcal{Q}(k) &= \texttt{WAIT}\ k; h \rightarrow l \rightarrow \texttt{STOP} \quad \text{for } k = 0, \ldots, N-1 \\ \mathcal{Q}(N) &= \texttt{STOP}\end{aligned}$$

$$\begin{aligned}T(k) &= l \rightarrow \texttt{SUCCESS}(k) \overset{1}{\rhd} T(k+1) \quad \text{for } k = 0, \ldots, N-1 \\ T(N) &= \texttt{SUCCESS}(N)\end{aligned}$$

- 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.