Kahn Process Networks

Gustav Cedersjö
Outline

What is a Kahn Process Network?
- Rules
- Example
- Properties

Formalizing Kahn Process Networks
- Channels
- Processes as Functions
- Processes as Monotonic Functions
- Example
  - Example
What is a Kahn Process Network?

Rules

- Process: Sequential program
- Communication: send(channel, value) and wait(channel)
- Transmission: Finite time
- FSM: compute $\Rightarrow$ wait
What is a Kahn Process Network?

Example

```
process f(In[Int] in₁, In[Int] in₂, Out[Int] out) {
    Bool b ← true
    repeat {
        Int i ← if b then wait(in₁) else wait(in₂)
        print(i)
        send(out, i)
        b ← ¬b
    }
}
```
What is a Kahn Process Network?

Properties

Parallelism  Processes may run in parallel.
Determinacy  The output of a process is determined by the history of input.
Composability A network of processes is also deterministic.
The history of a channel is a sequence $X^\omega$ of some domain $X$.

$$[-3, 5, 4, 4, 0] \in \mathbb{Z}^\omega$$

Sequences and the prefix relation $(X^\omega, \sqsubseteq)$ is a partially ordered set.

$$[h, e, l] \sqsubseteq [h, e, l, l, o]$$

$$[h, e, j] \not\sqsubseteq [h, e, l, l, o]$$

The prefix relation on pairs of sequences is defined as

$$(a, b) \sqsubseteq (a', b') \iff a \sqsubseteq a' \text{ and } b \sqsubseteq b'$$
Given a Kahn process and an input history there is only one possible output history.

- A sequential program is deterministic.
- No communication is allowed outside the channels.

A Kahn process is a functional mapping from input to output.

\[
p : X^\omega \rightarrow Y^\omega
\]

\[
f : (Z^\omega \times Z^\omega) \rightarrow Z^\omega
\]

\[
g : (X_1^\omega \times \cdots \times X_k^\omega) \rightarrow (Y_1^\omega \times \cdots \times Y_n^\omega)
\]
Adding more input at the end of an input sequence will only add more output at the end of the output sequence.

Kahn processes are prefix monotonic functional mappings. A function \( f : X^\omega \times Y^\omega \rightarrow Z^\omega \) is prefix monotonic if

\[
(x, y) \sqsubseteq (x', y') \Rightarrow f(x, y) \sqsubseteq f(x', y')
\]
Adding more input at the end of an input sequence will only add more output at the end of the output sequence.

Kahn processes are prefix monotonic functional mappings. A function \( f : X^\omega \times Y^\omega \rightarrow Z^\omega \) is prefix monotonic if

\[
(x, y) \sqsubseteq (x', y') \Rightarrow f(x, y) \sqsubseteq f(x', y')
\]
Formalizing Kahn Process Networks
Processes as Monotonic Functions

Adding more input at the end of an input sequence will only add more output at the end of the output sequence.

Kahn processes are prefix monotonic functional mappings. A function \( f : X^\omega \times Y^\omega \rightarrow Z^\omega \) is prefix monotonic if

\[
(x, y) \sqsubseteq (x', y') \Rightarrow f(x, y) \sqsubseteq f(x', y')
\]
Formalizing Kahn Process Networks
Processes as Monotonic Functions

Adding more input at the end of an input sequence will only add more output at the end of the output sequence.

Kahn processes are prefix monotonic functional mappings.
A function $f : X^\omega \times Y^\omega \rightarrow Z^\omega$ is prefix monotonic if

$$(x, y) \sqsubseteq (x', y') \Rightarrow f(x, y) \sqsubseteq f(x', y')$$
Adding more input at the end of an input sequence will only add more output at the end of the output sequence.

Kahn processes are prefix monotonic functional mappings. A function $f : X^\omega \times Y^\omega \rightarrow Z^\omega$ is prefix monotonic if

$$(x, y) \subseteq (x', y') \Rightarrow f(x, y) \subseteq f(x', y')$$
Formalizing Kahn Process Networks
Processes as Monotonic Functions

**Adding more input** at the end of an input sequence **will only add more output** at the end of the output sequence.

Kahn processes are prefix monotonic functional mappings. A function $f : X^\omega \times Y^\omega \rightarrow Z^\omega$ is prefix monotonic if

$$(x, y) \sqsubseteq (x', y') \Rightarrow f(x, y) \sqsubseteq f(x', y')$$
Kahn processes are prefix monotonic functions.

Monotonic functions compose to monotonic functions.

A Kahn process network is a prefix monotonic function.
Merge by taking first available token.
Example

Merge by taking first available token.
Merge by taking first available token.

This is not a function from just input to output sequences.
Merge by taking first available token.

This is not a function from just input to output sequences.
Formalizing Kahn Process Networks

Example

Merge by taking first available token.

\[ [1, 2] \]
\[ [4, 5] \]
\[ [4, 1, 5, 2] \]

This is not a function from just input to output sequences.
Formalizing Kahn Process Networks

Example

Merge by taking first available token.

\[ [1, 2, 3] \]
\[ [4, 5] \]
\[ [4, 1, 5, 2, 3] \]

\[ m \]

This is not a function from just input to output sequences.
Example

Merge by taking first available token.

\[ [1, 2, 3] \]
\[ [4, 5, 6] \]
\[ m \]
\[ [4, 1, 5, 2, 3, 6] \]

This is not a function from just input to output sequences.
Merge by taking first available token.

This is not a function from just input to output sequences.
Merge by taking first available token.

Example

This is not a function from just input to output sequences.
Merge by taking first available token.

\[
\begin{align*}
[1, 2, 3] & \quad [4, 1, 5, 2, 3, 6] \\
[4, 5, 6] & \\
\end{align*}
\]

This is not a function from just input to output sequences.
Formalizing Kahn Process Networks

Example

Merge by taking first available token.

This is not a function from just input to output sequences.
Merge by taking first available token.

This is not a function from just input to output sequences.
Merge by taking first available token.

This is not a function from just input to output sequences.
Formalizing Kahn Process Networks

Example

Merge by taking first available token.

This is not a function from just input to output sequences.
Reverse the input sequence.
Reverse the input sequence.

\[
\begin{align*}
[1, 2, 3] & \rightarrow r & [3, 2, 1] \\
\[ & \rightarrow r & \]
\end{align*}
\]
Reverse the input sequence.

\[ [1, 2, 3] \rightarrow r \rightarrow [3, 2, 1] \]

\[ [1, 2, 3, 4] \rightarrow r \rightarrow [4, 3, 2, 1] \]
Reverse the input sequence.

This is not a prefix monotonic function.