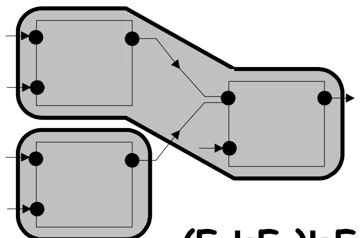
An Interface Algebra for Task Graphs

Slobodan Matic   Tom Henzinger

UC Berkeley, EPFL

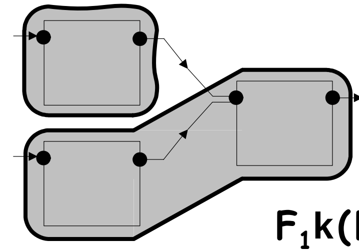
Interface-based Analysis

- Interface composition properties
  - Incremental design



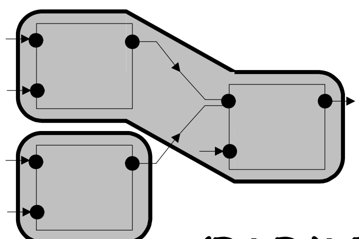
$(F_1 k F_3) k F_2$

$\stackrel{?}{=}$



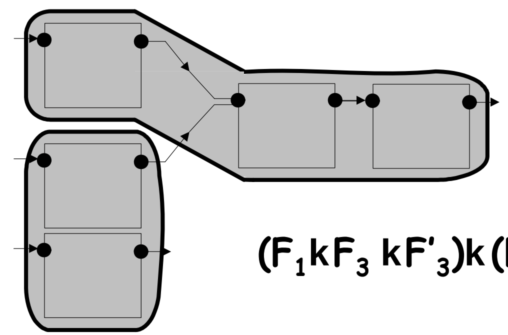
$F_1 k (F_2 k F_3)$

  - Independent refinement



$(F_1 k F_3) k F_2$

o

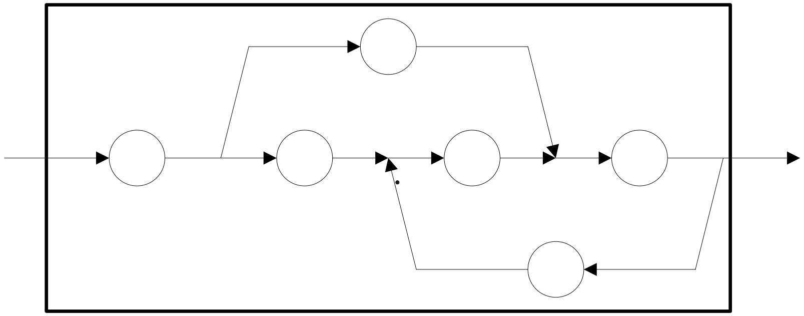


$(F_1 k F_3\ k F'_3) k (F_2 k F'_2)$

Component Model

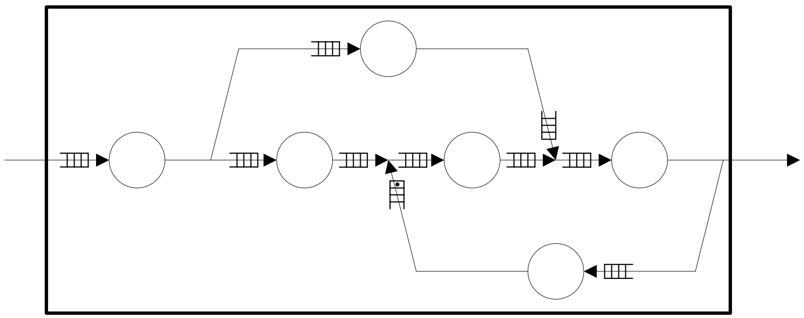
Task graph

- data precedence
- cycles allowed
- one token consumed and produced



Component Model

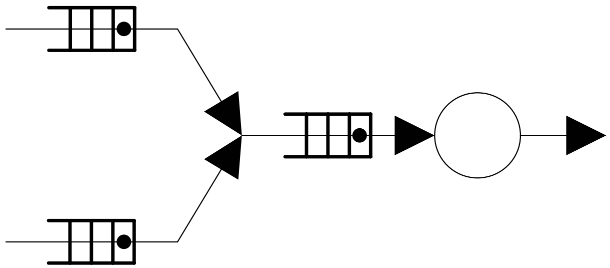
Task graph
- data precedence
- cycles allowed
- one token consumed and produced



Task communication
- input FIFO buffers

Tasks with multiple-inputs
- AND type

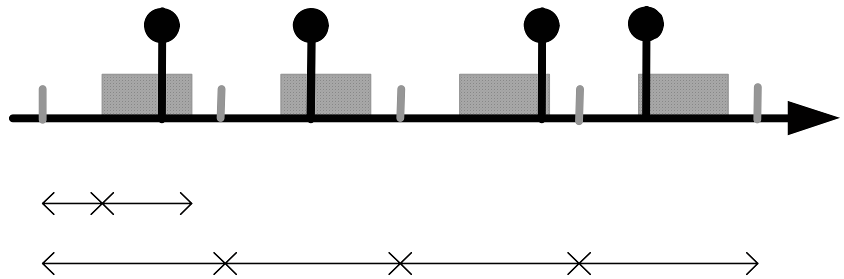
Cyclic dependencies
- initial tokens on each cycle



Event model

Periodic with jitter

- single rate, period p



Given

- period **p**
- jitter **j**
- phase **t**

stream e **2** L(**p**,**j**,**t**) iff for all k

t + k**p** · e(k) · **t** + k**p** + **j**

Event model

Periodic with jitter

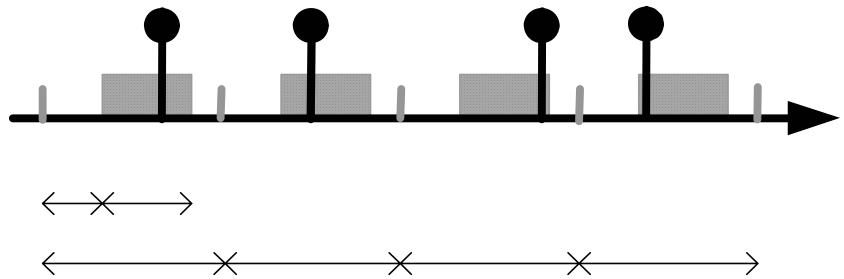
- single rate p

Given

- period **p**
- jitter **j**
- phase **t**

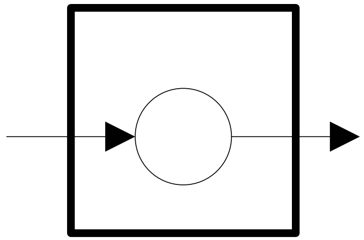
stream e **2** L(**p**,**j**,**t**) iff for all k

t + k**p** · e(k) · **t** + k**p** + **j**



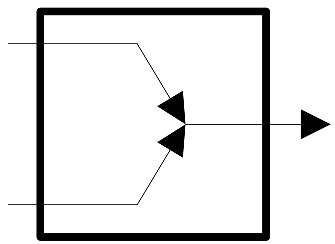
Event propagation

Task response-time bounds [l,u]

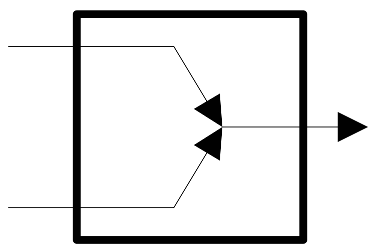


$t_o = t_i + l$
$j_o = j_i + (u\text{-}l)$

AND element

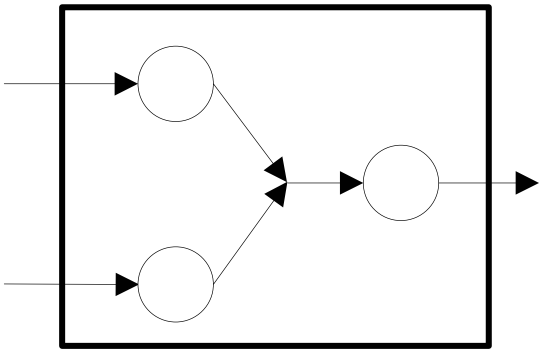


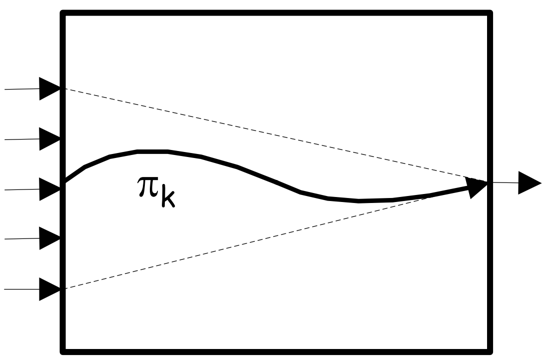
a)
$j_o = \max(j_1, j_2)$ (Jersak04)



b)
$t_o = \max(t_1, t_2)$
$j_o = \max(j_1+t_1, j_2+t_2) - t_0$
$= \max(j_1+t_1, j_2+t_2) - \max(t_1, t_2)$
$\cdot \max(j_1, j_2)$

Acyclic Graphs





a)

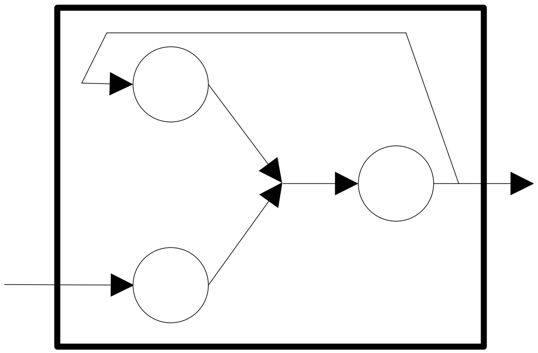
$$j_o = \max_{\pi_k} (j_k + \sum (u-l)) > j_i$$

b)

$$t_o = \max_{\pi_k} (t_k + \sum l)$$

$$j_o = \max_{\pi_k} (t_k + j_k + \sum u) - t_o$$

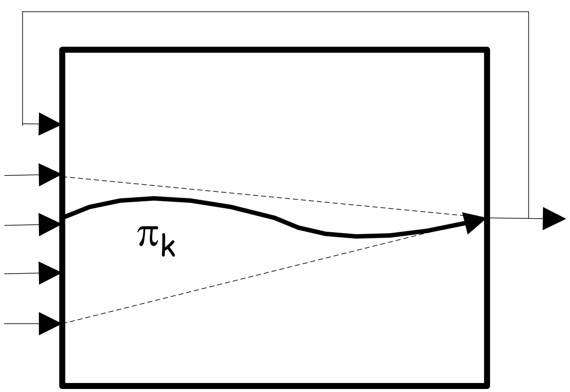
Cyclic Graphs



Fixed point equations

$t_o = t_1 + p$

$j_o = j_1$



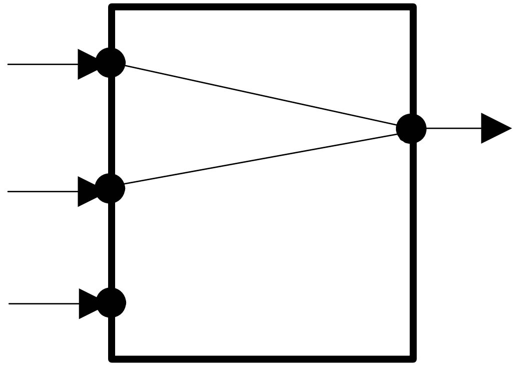
b)

$$t_o = \max_{\pi_k,\ k \neq 1} (t_k + \sum l)$$

$$j_o = \max_{\pi_k,\ k \neq 1} (t_k + j_k + \sum u) - t_o$$

1

Interface



Assumption
Environment provides inputs that satisfy

Input predicate P_I

i_1 **2** $L(p, j_1, t_1)$

Æ i_2 **2** $L(p, j_2, t_2)$

Æ r ¸ c

Guarantee
Component produces outputs that satisfy

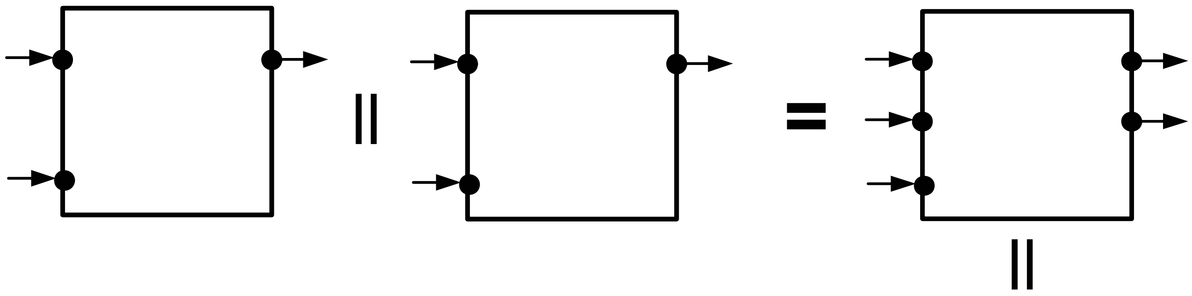
Output predicate P_O

o **2** $L(p, j_o, t_o)$

Interface Algebra

Operations

- Composition
- Connection
- Join



Interface Algebra

Operations

- Composition
- Connection
- Join

Interface Algebra

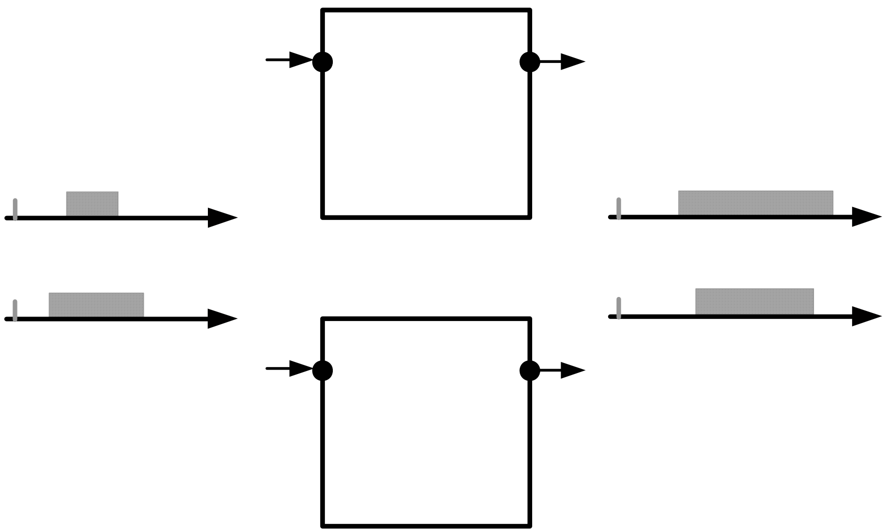
Operations

- Composition
- Connection
- Join

Refinement relation

F_1 refines F_2 $\quad (F_1\ ^1\ F_2)$

$P_I(F_2)$) $P_I(F_1)$ and $P_O(F_1)$) $P_O(F_2)$



Algebra Properties

Incremental design

Associative algebra operations (flexibility)

If $(F\ op_1\ G)\ op_2\ H$ and $G\ op_2\ H$ are defined
then $F\ op_1\ (G\ op_2\ H)$ is def. and $(F\ op_1\ G)\ op_2\ H = F\ op_1\ (G\ op_2\ H)$

If $(F\ op_1\ G)\ op_2\ H$ and $F\ op_2\ H$ are defined
then $(F\ op_2\ H)\ op_1\ G$ is def. and $(F\ op_1\ G)\ op_2\ H = (F\ op_2\ H)\ op_1\ G$

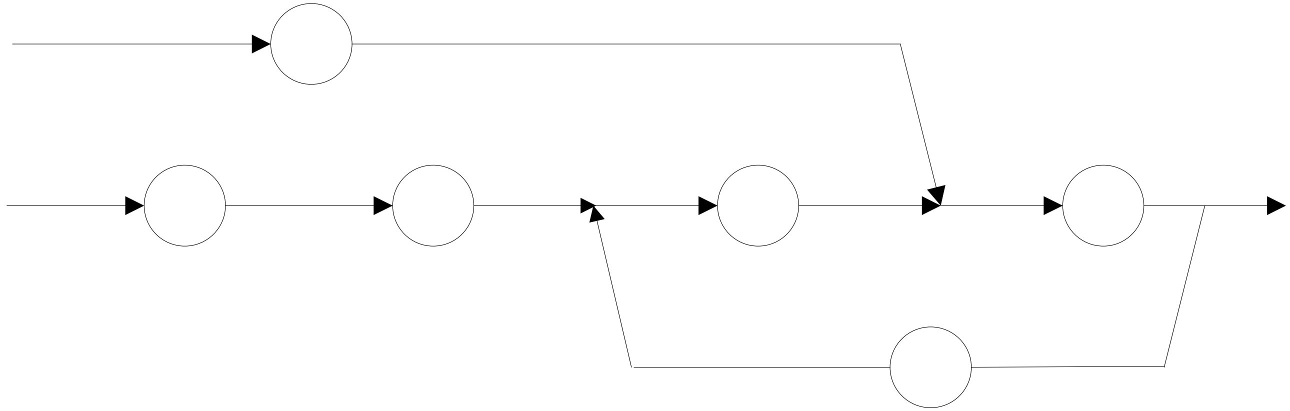
Independent refinement

Refinement preserved under operations (no global checks)

If $F\ op\ G$ is def. and $F' \ 1\ F$
then $F'\ op\ G$ is def. and $(F'\ op\ G)\ 1\ (F\ op\ G)$

If $F'_j\ 1\ F_j$ $(j=1,\ldots,n)$
then $E(F'_1,\ldots,F'_n)\ 1\ E(F_1,\ldots,F_n)$

Example

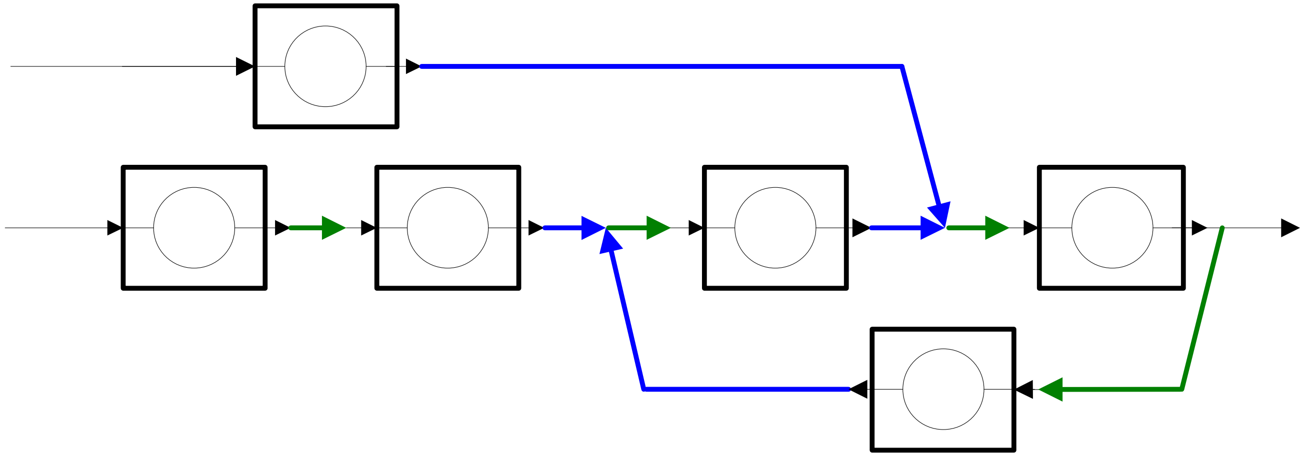


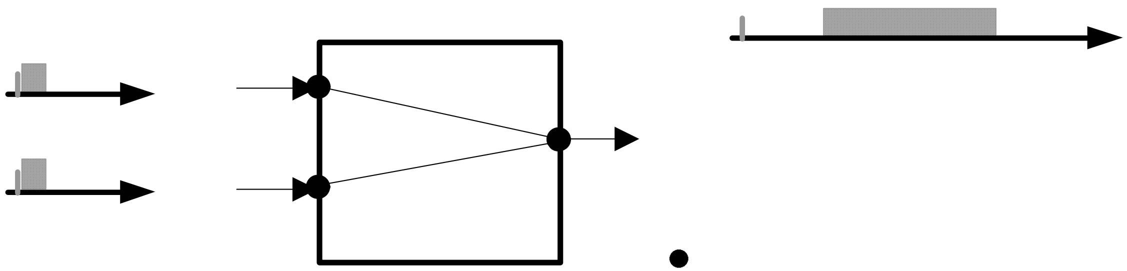
- Allocation: 1,2,3 to R_1
  4,5,6 to R_2
- For all tasks execution-time bounds [2,3]
- Scheduling: fixed-priority
  lower number higher priority
- p=20

$[t_1,j_1]=[t_2,j_2]=[0,1]$

i_1

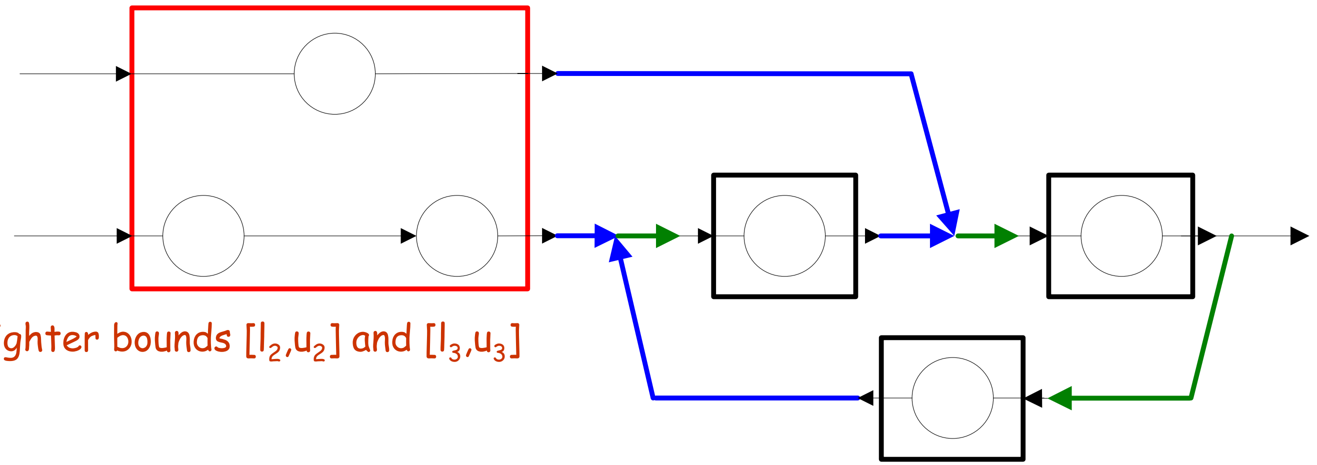
Composition F_1

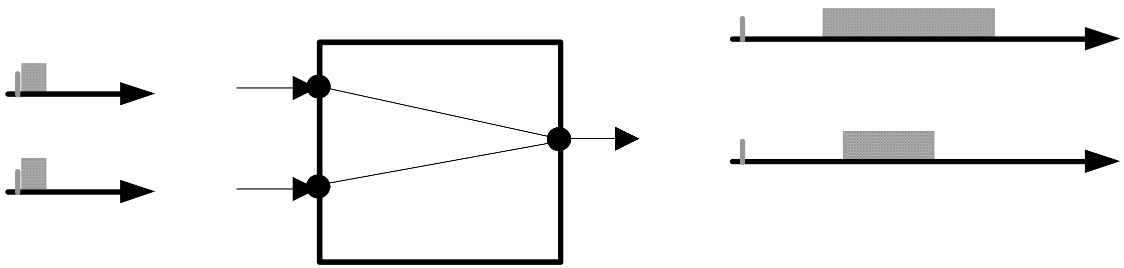




i_1 $\frac{1}{0}$

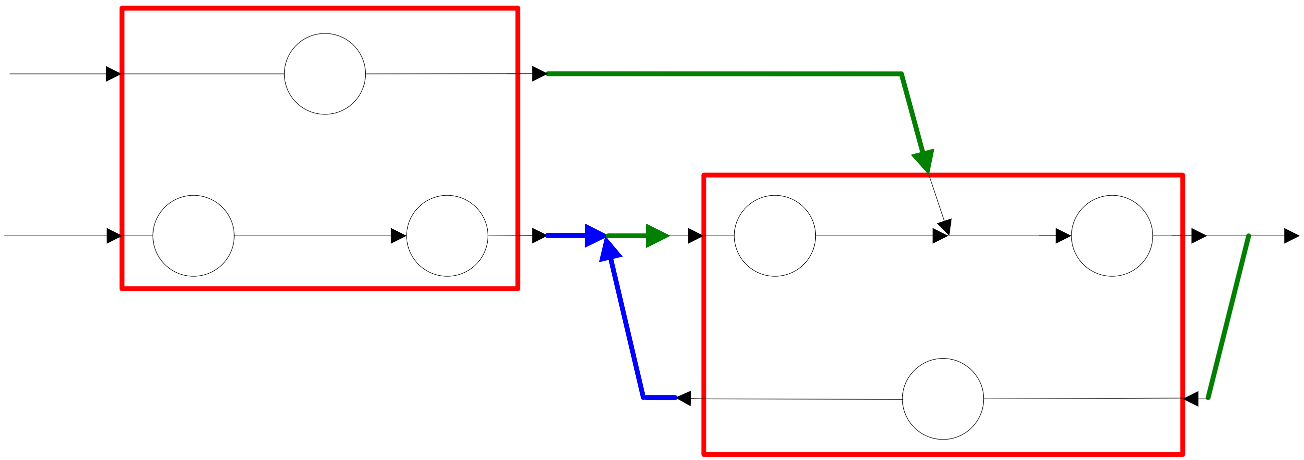
Composition F_2

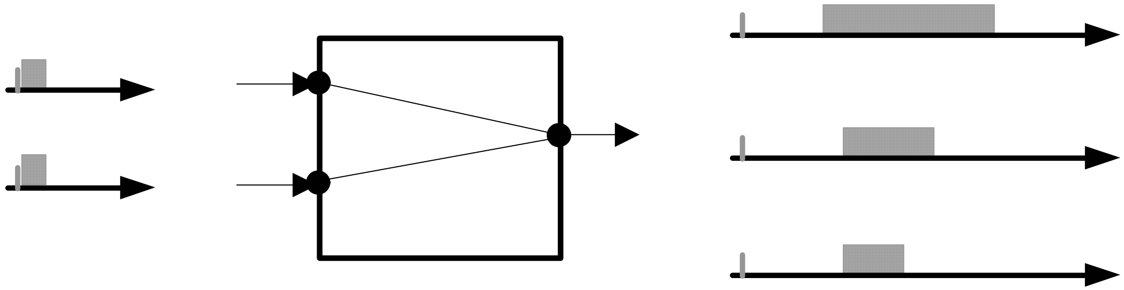




i_1 1 F_2 1 F_1 0

Composition F_3





i 1 1 0

F_3 1 F_2 1 F_1

Summary

- Phase information required for cyclic task graphs with multiple inputs
- Incremental design and independent refinement preserved

- Multiple rate graphs
- More general stream variability characterization