Distributed Adaptation of Dining Philosophers

Luuk Groenewegen

Leiden University

FACS 2010
Guimarães, October 14-16, 2010

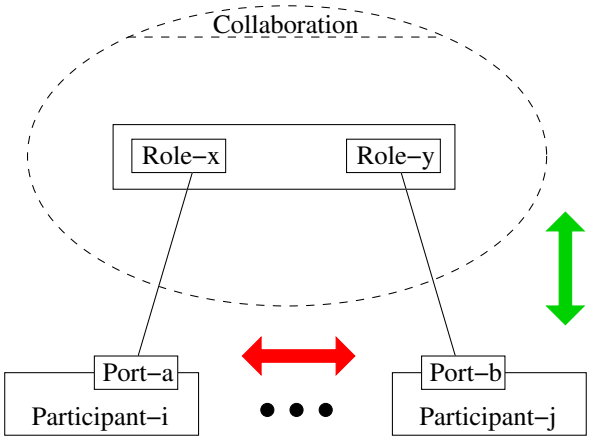
joint work with Suzana Andova and Erik de Vink

extended presentation version

Outline

- Problem situation
- Idea of solution: coordination
- Solution for unforeseen coordination
- Conclusion

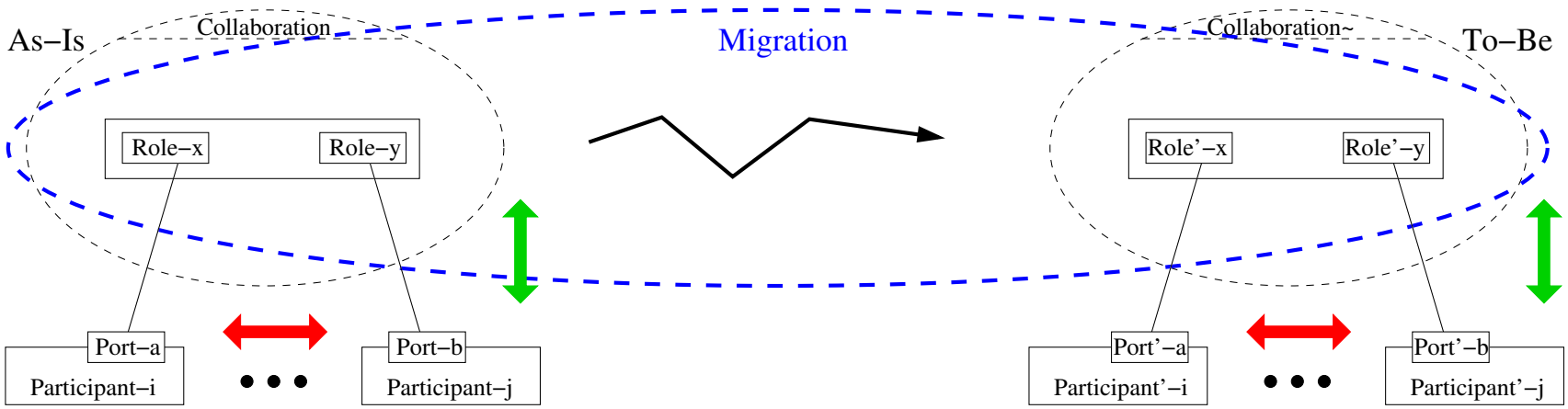
UML-like collaboration diagram



Coordination problem: dynamic consistency

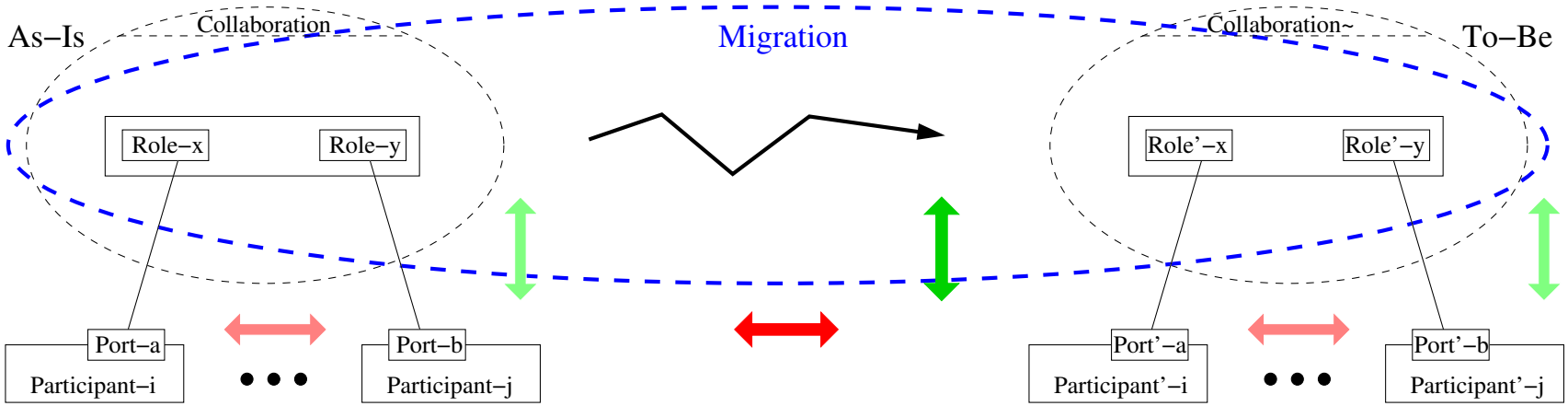
- vertical dynamic consistency
- horizontal dynamic consistency

UML-like collaboration diagrams during migration



Adaptation problem: dynamic consistency again

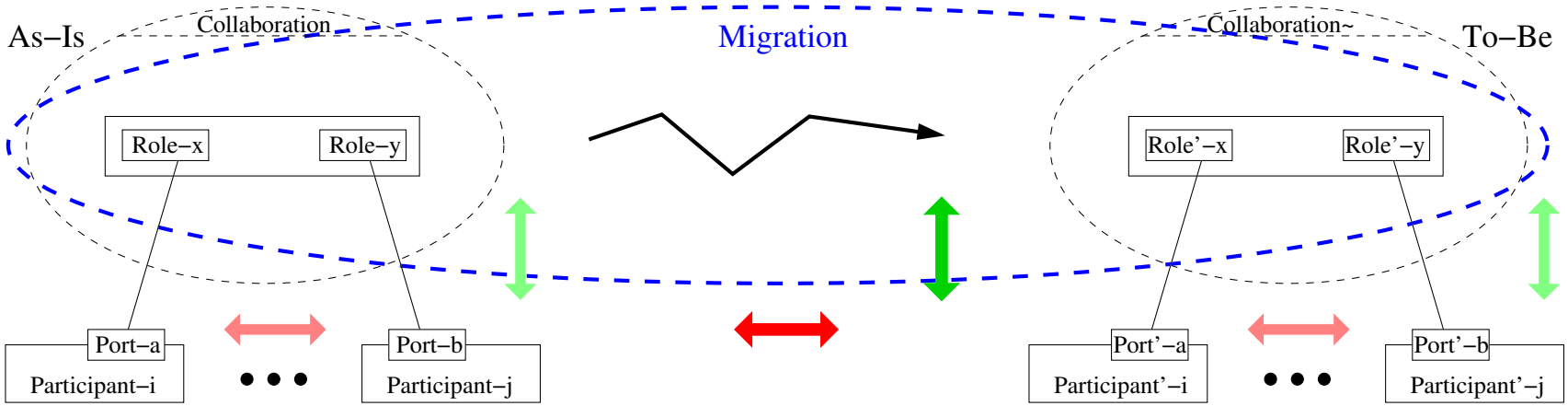
UML-like collaboration diagrams during migration



Adaptation problem: dynamic consistency again

- during migration: on-the-fly / no quiescence

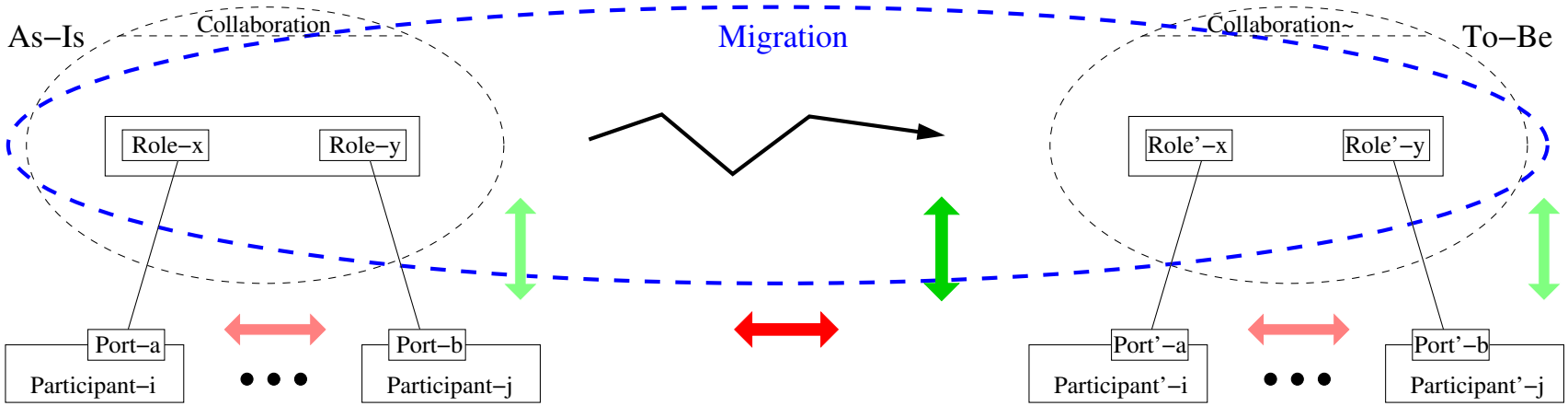
UML-like collaboration diagrams during migration



Adaptation problem: dynamic consistency again

- during migration: on-the-fly / no quiescence
- distributed

UML-like collaboration diagrams during migration



Adaptation problem: dynamic consistency again

- during migration: on-the-fly / no quiescence
- distributed
- formal analysis of migration trajectories

coordination and adaptation

normal, foreseen coordination can be specified

- through dynamicity of temporary constraints

coordination and adaptation

normal, foreseen coordination can be specified

- through dynamicity of temporary constraints

intuitive observation:

adaptation is like unforeseen coordination

- full As-Is has starting constraints
- full To-Be has final constraints

coordination and adaptation

normal, foreseen coordination can be specified

- through dynamicity of temporary constraints

intuitive observation:

adaptation is like unforeseen coordination

- full As-Is has starting constraints
- full To-Be has final constraints
- in between are intermediate constraints

coordination and adaptation

coordination language Paradigm:

with respect to any participant of a collaboration

- constraint imposed on behaviour: *phase*
- constraint committed to: *trap* of a phase

coordination and adaptation

coordination language Paradigm:

with respect to any participant of a collaboration

- constraint imposed on behaviour: *phase*
- constraint committed to: *trap* of a phase
- the language Paradigm guarantees vertical consistency through its syntax

coordination and adaptation

migration coordination of any Paradigm model towards another Paradigm model is carried out via Paradigm pattern / component *McPal*

coordination and adaptation

migration coordination of any Paradigm model towards another Paradigm model is carried out via Paradigm pattern / component *McPal*

- JIT modeling of new adaptation as extension
- while keeping As-Is constraints

coordination and adaptation

migration coordination of any Paradigm model towards another Paradigm model is carried out via Paradigm pattern / component *McPal*

- JIT modeling of new adaptation as extension
- while keeping As-Is constraints
- then, migrating as extension specifies

coordination and adaptation

migration coordination of any Paradigm model towards another Paradigm model is carried out via Paradigm pattern / component *McPal*

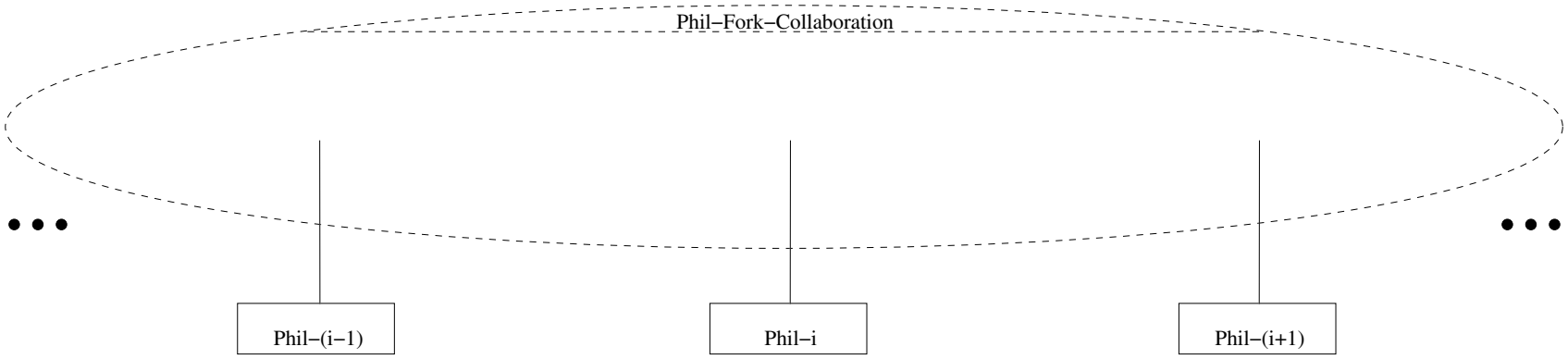
- JIT modeling of new adaptation as extension
- while keeping As-Is constraints
- then, migrating as extension specifies
- last, shrinking from extension to To-Be

animation of Dining Philosophers

an example

5 *Phil*osophers and 5 *Fork*s

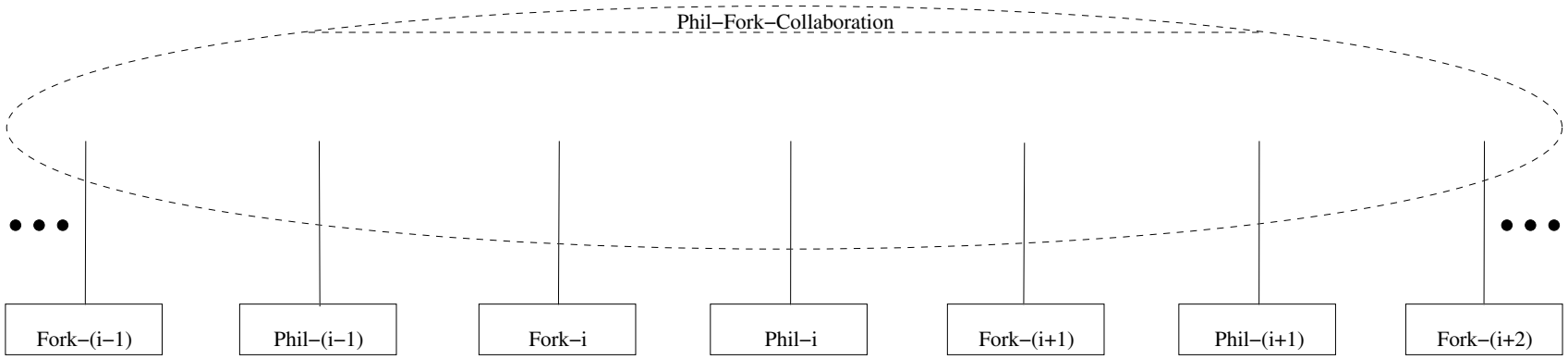
animation of Dining Philosophers



Paradigm model specifies coordination between

- 5 $Phil_i$ participants

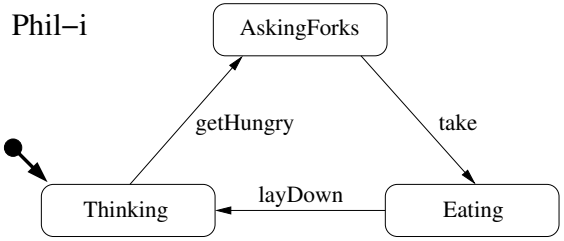
animation of Dining Philosophers



Paradigm model specifies coordination between

- 5 $Phil_i$ participants
- 5 $Fork_i$ participants
- each $Fork_i$ shared between $Phil_{i-1}$ and $Phil_i$

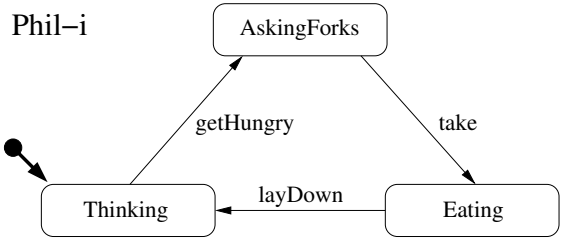
animation of Dining Philosophers

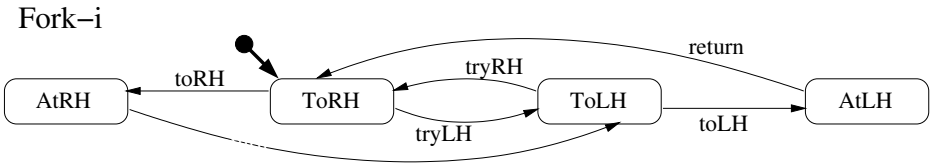


Paradigm model coordinates

- behaviour of 5 $Phil_i$

animation of Dining Philosophers

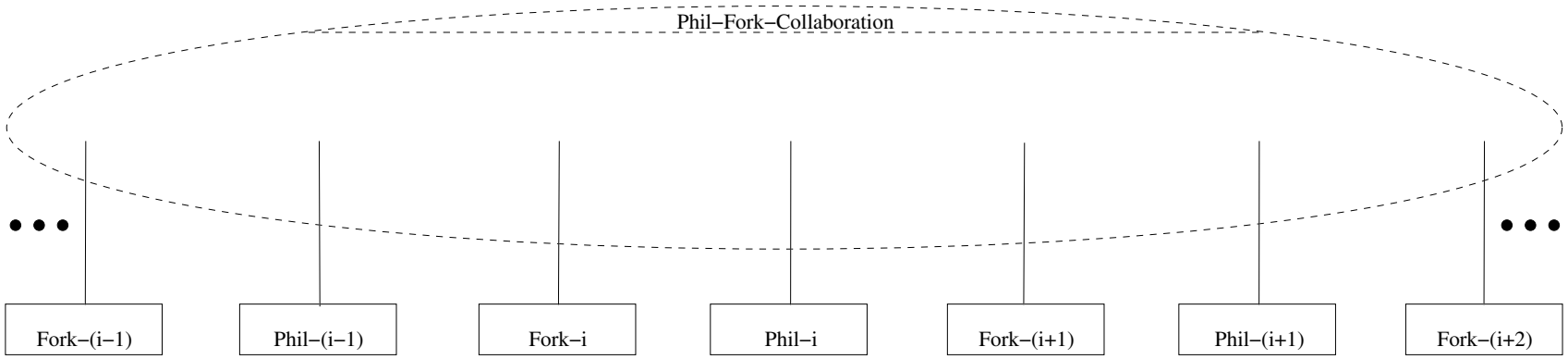




Paradigm model coordinates

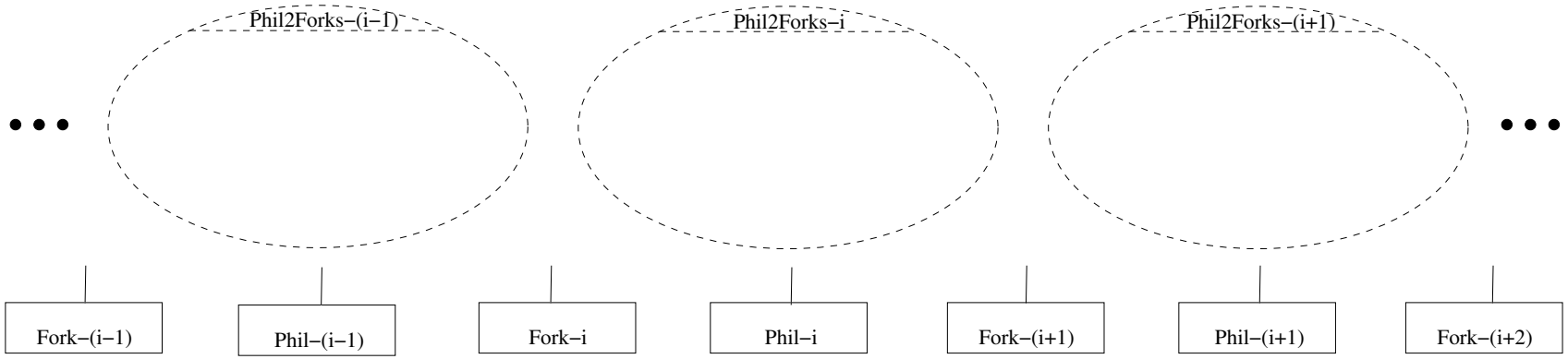
- behaviour of 5 $Phil_i$
- behaviour of 5 $Fork_i$

animation of Dining Philosophers



Paradigm model organizes *Phil*–*Fork* collaboration

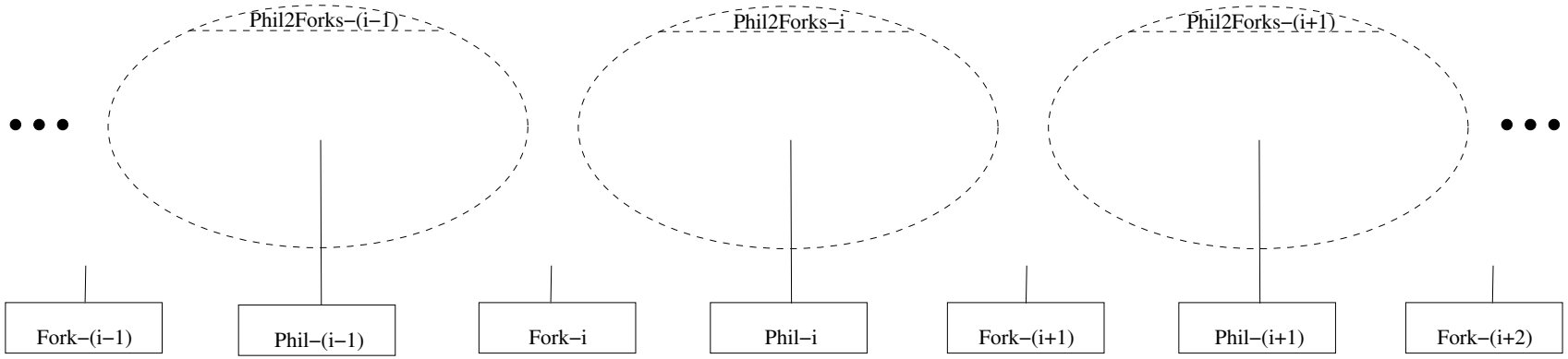
animation of Dining Philosophers



Paradigm model organizes *Phil*–*Fork* collaboration

- as 5 distributed collaborations $Phil2Forks_i$

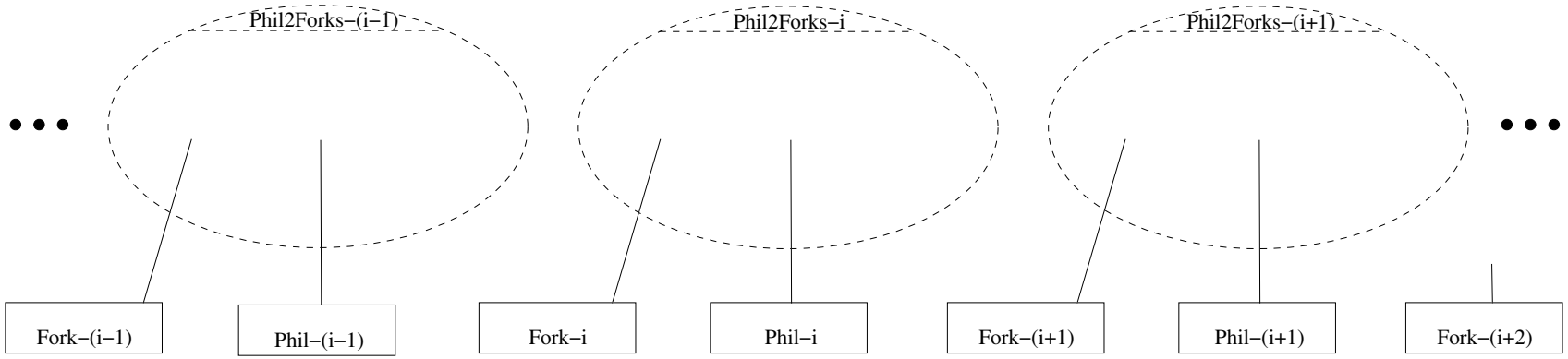
animation of Dining Philosophers



Paradigm model organizes *Phil*–*Fork* collaboration

- as 5 distributed collaborations $Phil2Forks_i$
- each having as participants: $Phil_i$

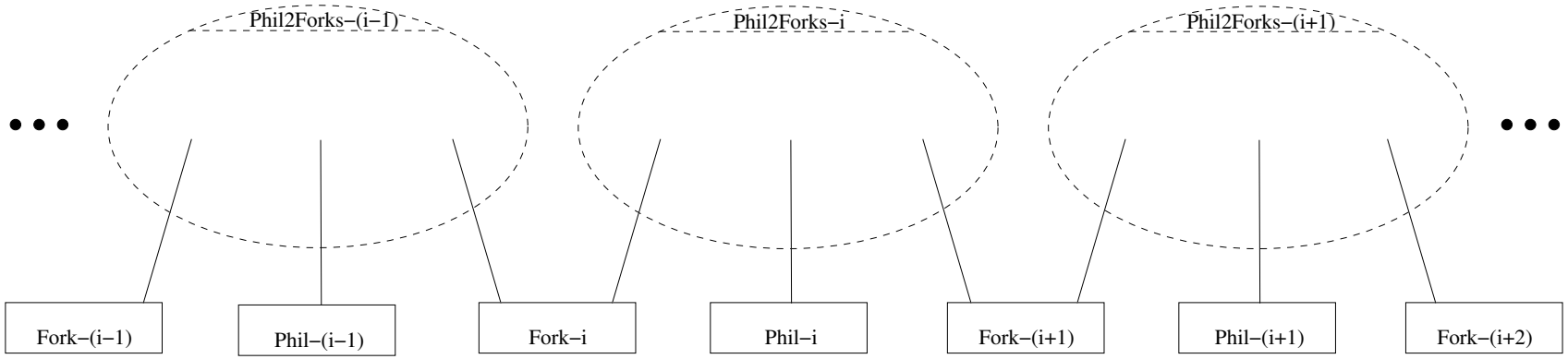
animation of Dining Philosophers



Paradigm model organizes *Phil*–*Fork* collaboration

- as 5 distributed collaborations $Phil2Forks_i$
- each having as participants: $Phil_i$, $Fork_i$

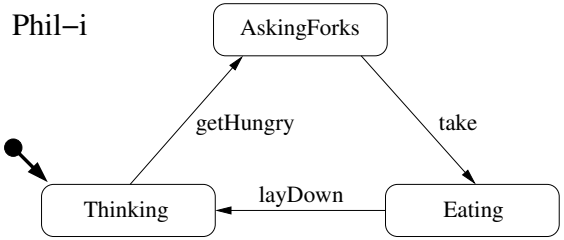
animation of Dining Philosophers



Paradigm model organizes *Phil*–*Fork* collaboration

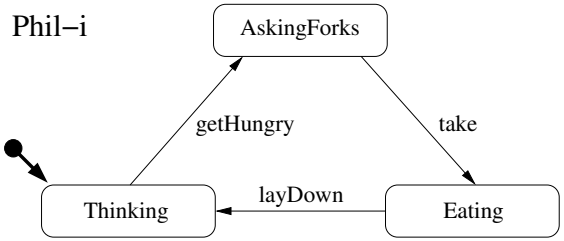
- as 5 distributed collaborations $Phil2Forks_i$
- each having as participants: $Phil_i$, $Fork_i$, $Fork_{i+1}$

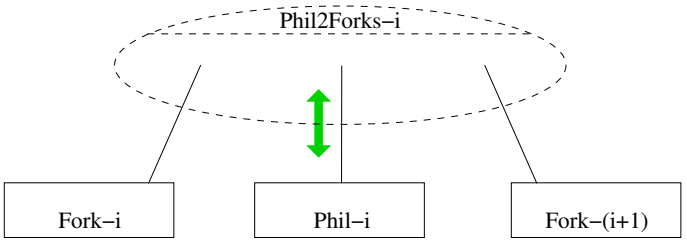
animation of Dining Philosophers



Phil$_i$ behaviour is coordinated

animation of Dining Philosophers

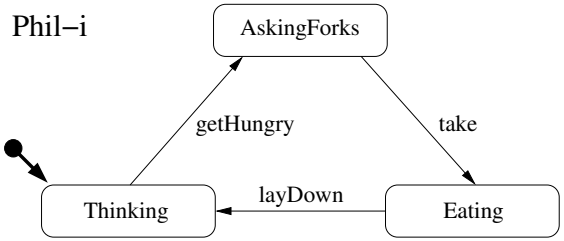


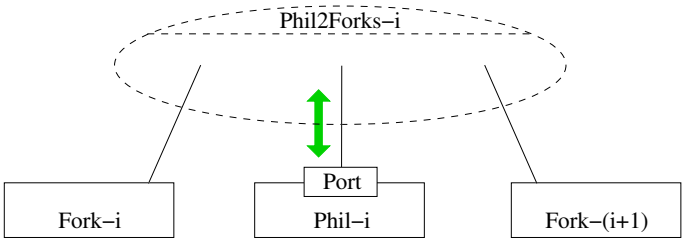


Phil$_i$ behaviour is coordinated

- by constraining it

animation of Dining Philosophers

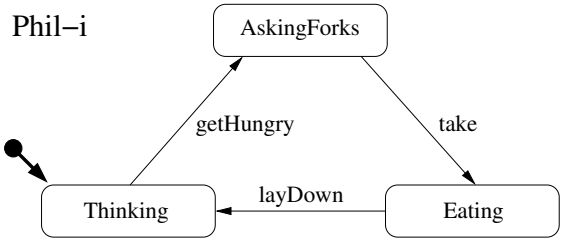


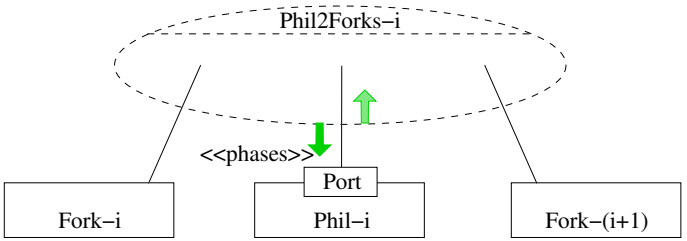


Phil$_i$ behaviour is coordinated

- by constraining it at its port

animation of Dining Philosophers

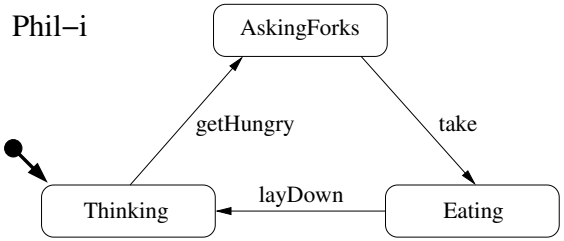


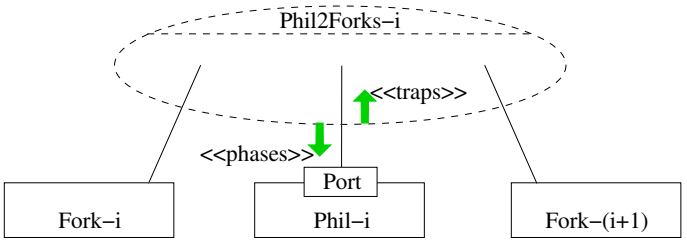


Phil$_i$ behaviour is coordinated

- by constraining it at its port
- via imposing phases on it

animation of Dining Philosophers

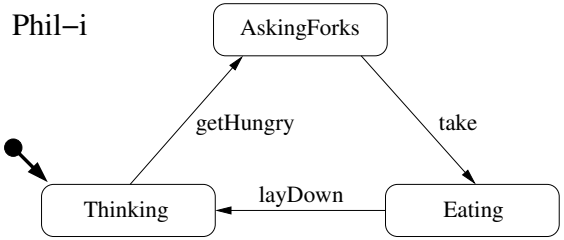




Phil$_i$ behaviour is coordinated

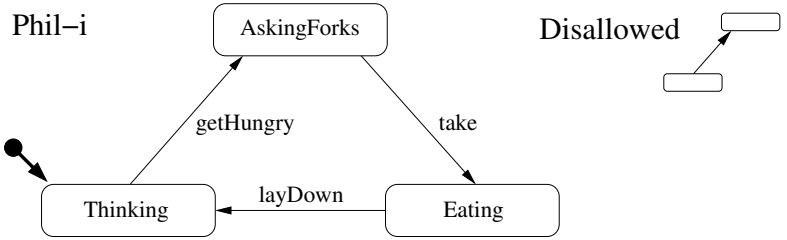
- by constraining it at its port
- via imposing phases on it
- via committing to traps by it

animation of Dining Philosophers



Paradigm model constrains $Phil_i$ behaviour

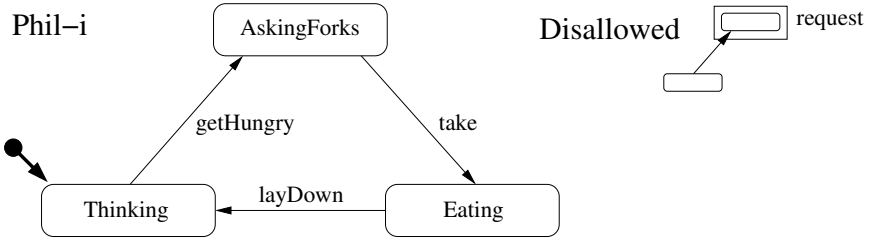
animation of Dining Philosophers



Paradigm model constrains *Phil*$_i$ behaviour via

- phase *Disallowed*: no eating

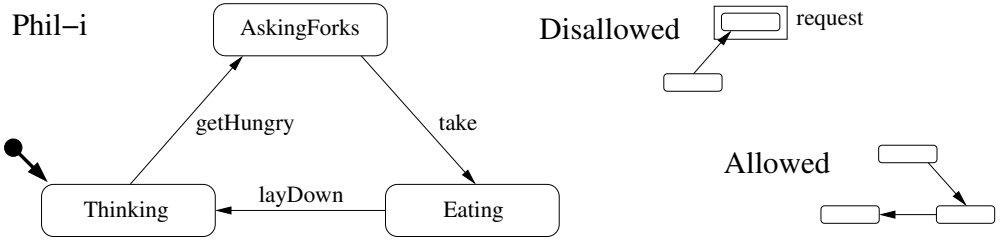
animation of Dining Philosophers



Paradigm model constrains $Phil_i$ behaviour via

- phase *Disallowed*: no eating
- trap *request*: wanting to eat

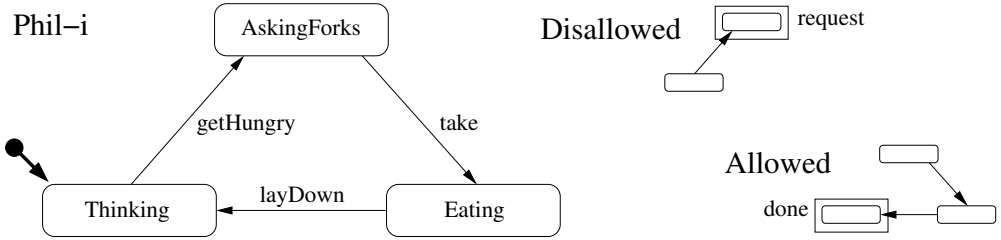
animation of Dining Philosophers



Paradigm model constrains *Phil*$_i$ behaviour via

- phase *Disallowed*: no eating
- trap *request*: wanting to eat
- phase *Allowed*: eating

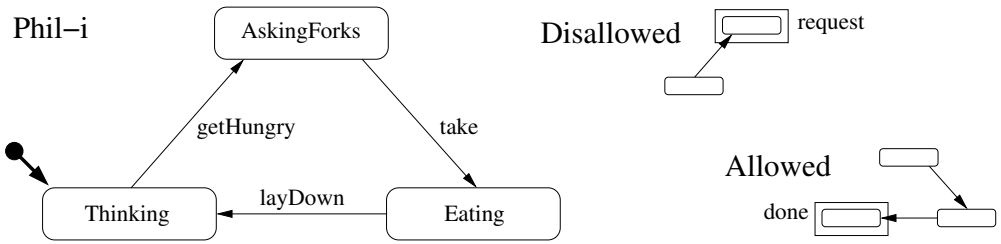
animation of Dining Philosophers



Paradigm model constrains $Phil_i$ behaviour via

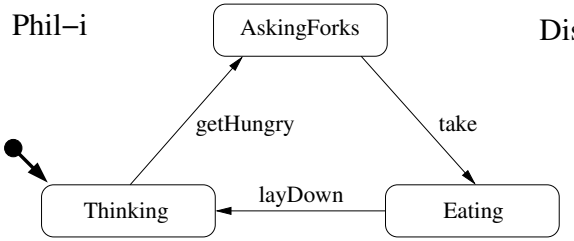
- phase *Disallowed*: no eating
- trap *request*: wanting to eat
- phase *Allowed*: eating
- trap *done*: giving up eating

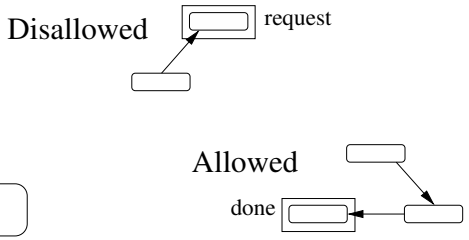
animation of Dining Philosophers

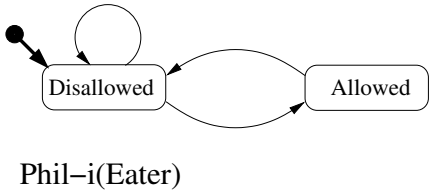


Paradigm model gives $Phil_i$'s constraint dynamicity through role $Phil_i(Eater)$

animation of Dining Philosophers



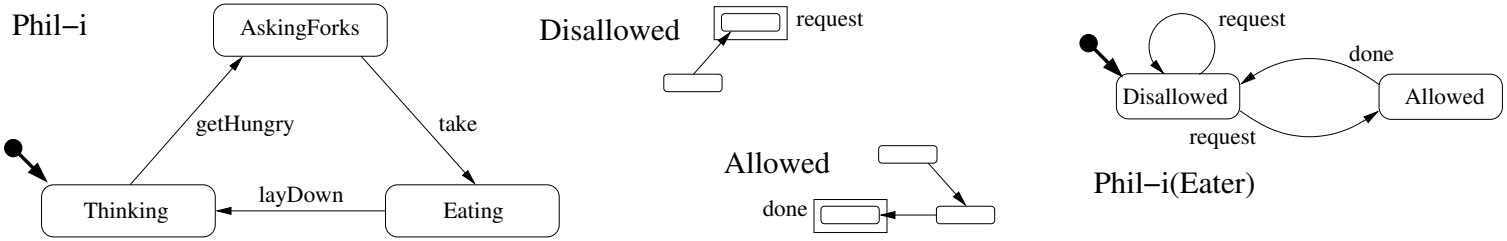




Paradigm model gives $Phil_i$'s constraint dynamicity through role $Phil_i(Eater)$

- phases are role states

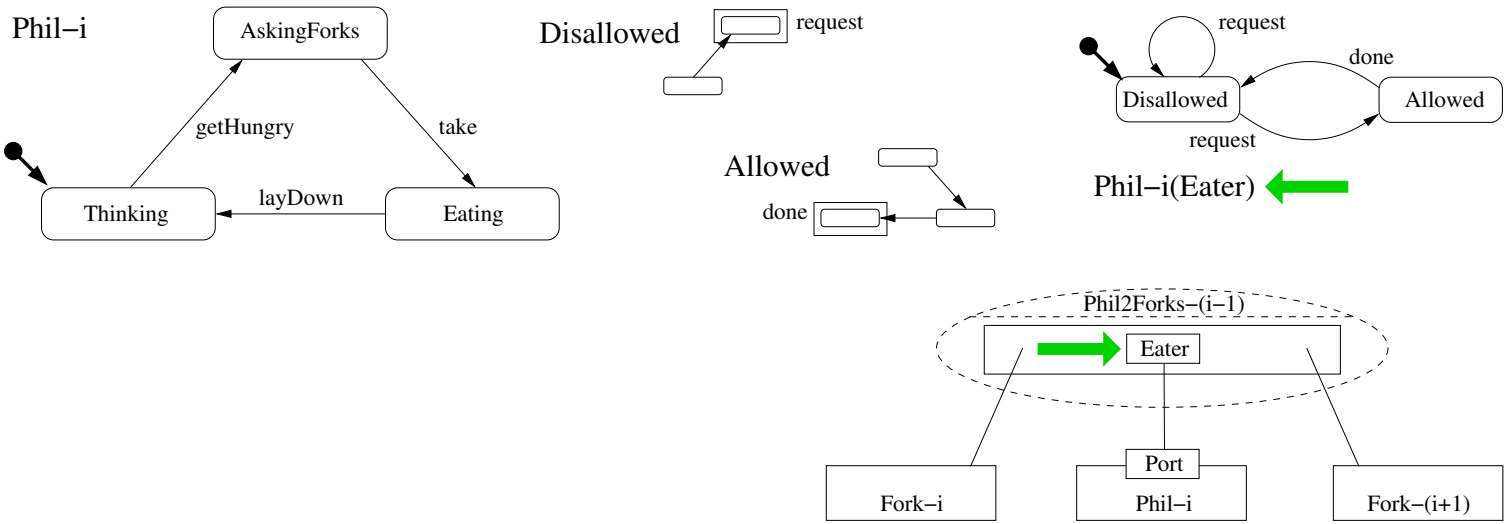
animation of Dining Philosophers



Paradigm model gives $Phil_i$'s constraint dynamicity through role $Phil_i(Eater)$

- phases are role states
- *connecting* traps label phase transfers

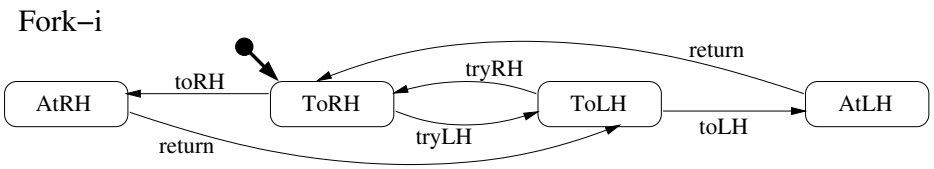
animation of Dining Philosophers



Paradigm model gives $Phil_i$'s constraint dynamicity through role $Phil_i(Eater)$

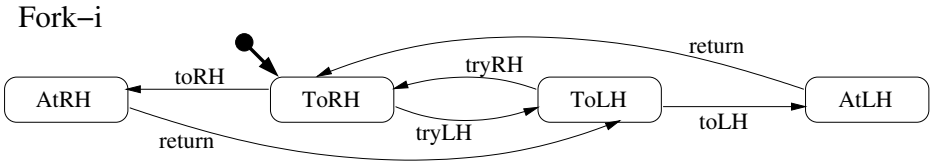
- phases are role states
- *connecting* traps label phase transfers

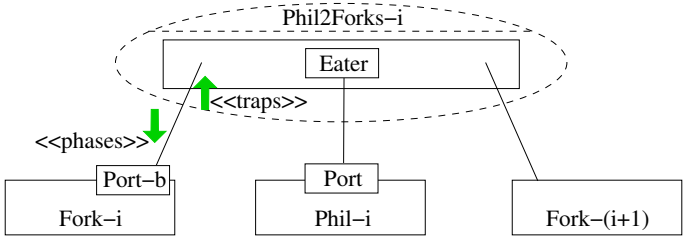
animation of Dining Philosophers



wrt $Phil_i$'s left hand:

animation of Dining Philosophers

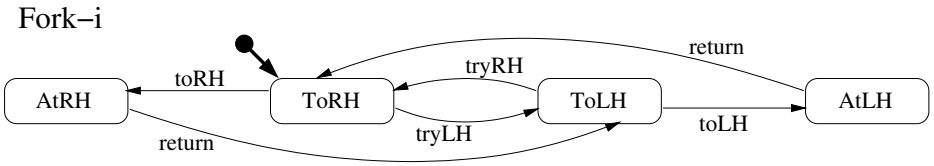




wrt $Phil_i$'s left hand: constraints on $Fork_i$

- arrive at its port-b

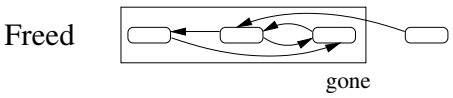
animation of Dining Philosophers

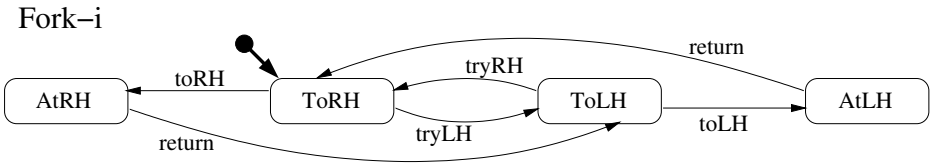


wrt $Phil_i$'s left hand: constraints on $Fork_i$

- arrive at its port-b
- are: phase *Freed*

animation of Dining Philosophers

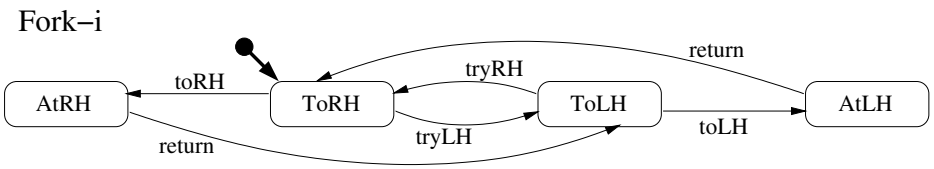


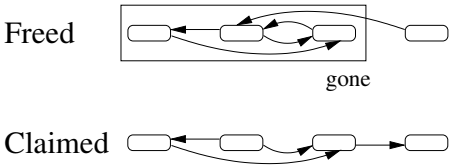


wrt $Phil_i$'s left hand: constraints on $Fork_i$

- arrive at its port-b
- are: phase *Freed* ; trap *gone*

animation of Dining Philosophers

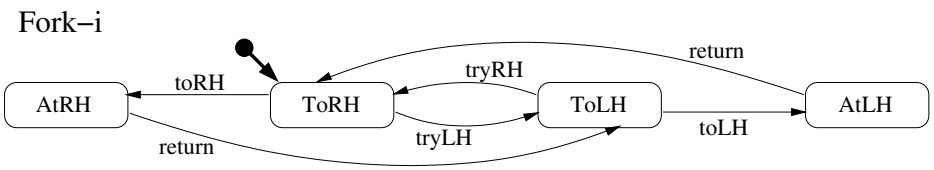


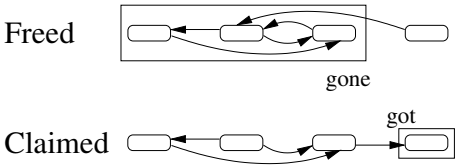


wrt *Phil*$_i$'s left hand: constraints on *Fork*$_i$

- arrive at its port-b
- are: phases *Freed*, *Claimed*; trap *gone*

animation of Dining Philosophers

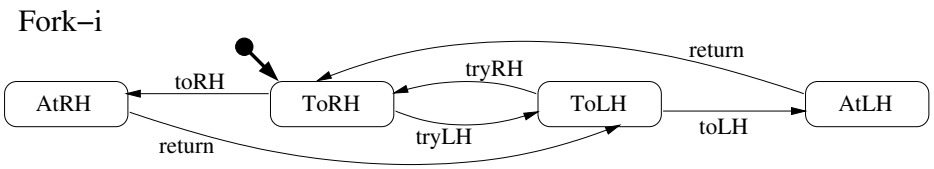


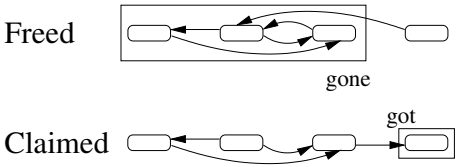


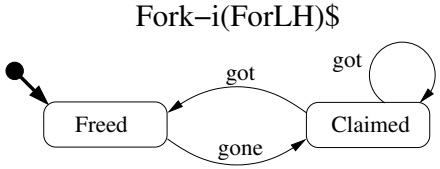
wrt *Phil*$_i$'s left hand: constraints on *Fork*$_i$ are

- arrive at its port-b
- are: phases *Freed*, *Claimed*; traps *gone*, *got*

animation of Dining Philosophers



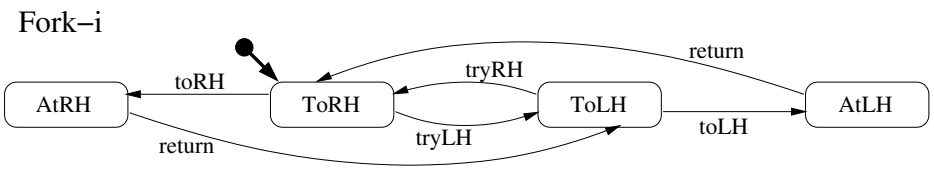


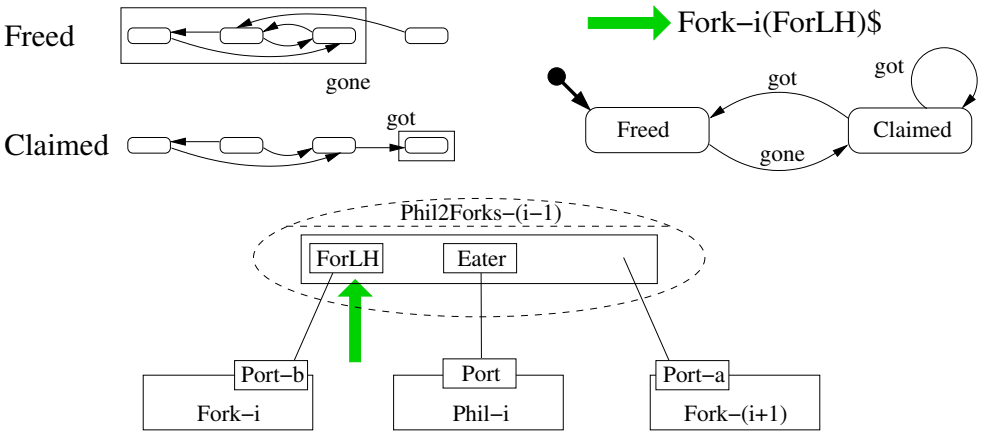


wrt *Phil*$_i$'s left hand: constraints on *Fork*$_i$

- arrive at its port-b
- are: phases *Freed*, *Claimed*; traps *gone*, *got*
- and via role *Fork*$_i$(*ForLH*)

animation of Dining Philosophers

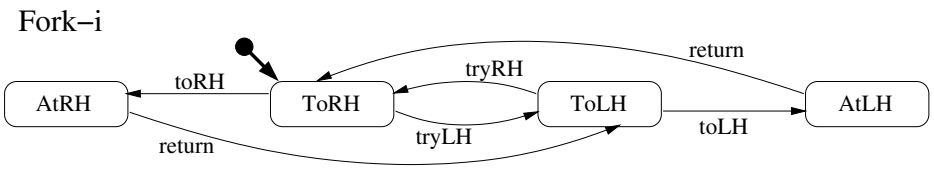


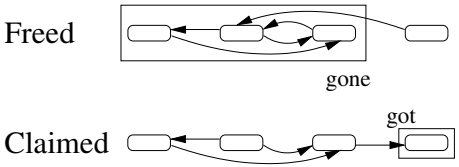


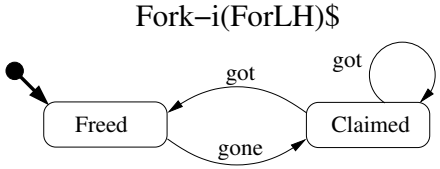
wrt $Phil_i$'s left hand: constraints on $Fork_i$

- arrive at its port-b
- are: phases *Freed*, *Claimed*; traps *gone*, *got*
- and are governed via role $Fork_i(ForLH)$

animation of Dining Philosophers

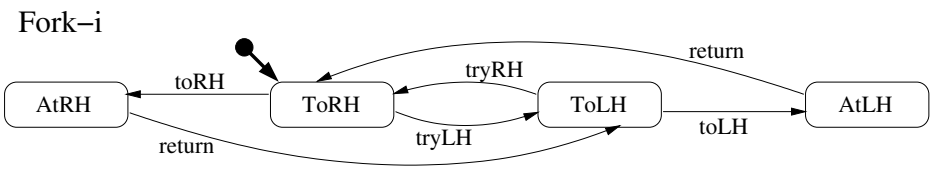


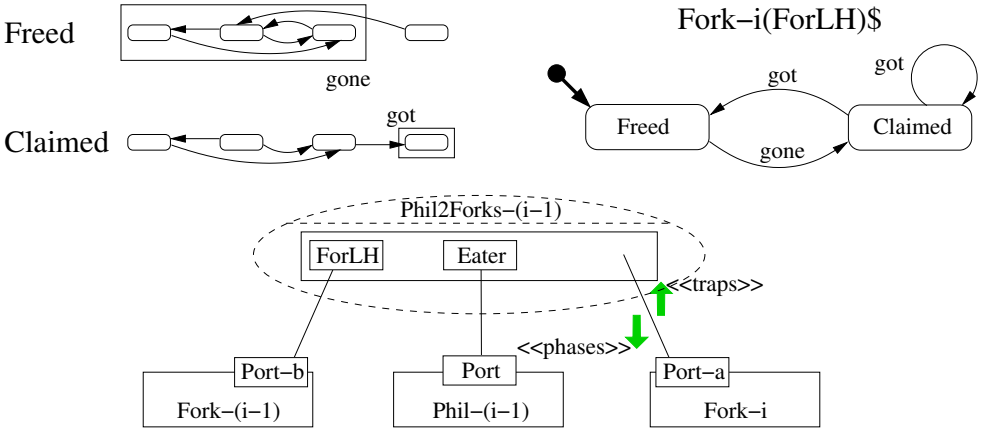




wrt $Phil_{i-1}$'s right hand:

animation of Dining Philosophers

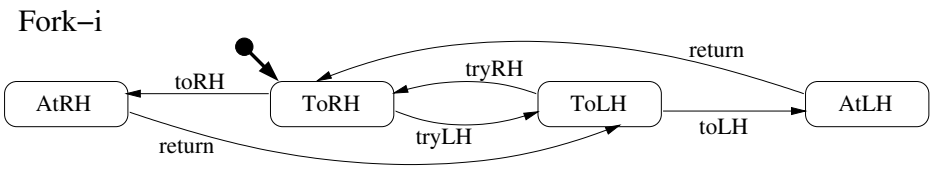


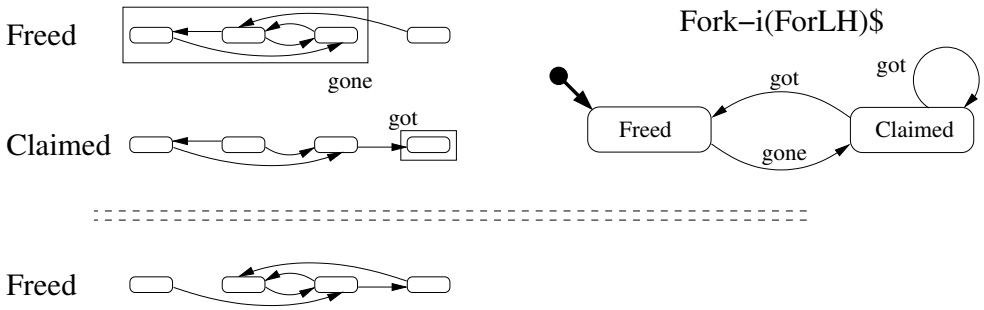


wrt $Phil_{i-1}$'s right hand: constraints on $Fork_i$

- arrive at its port-a

animation of Dining Philosophers

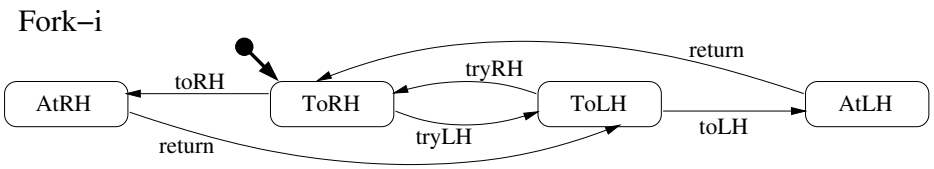


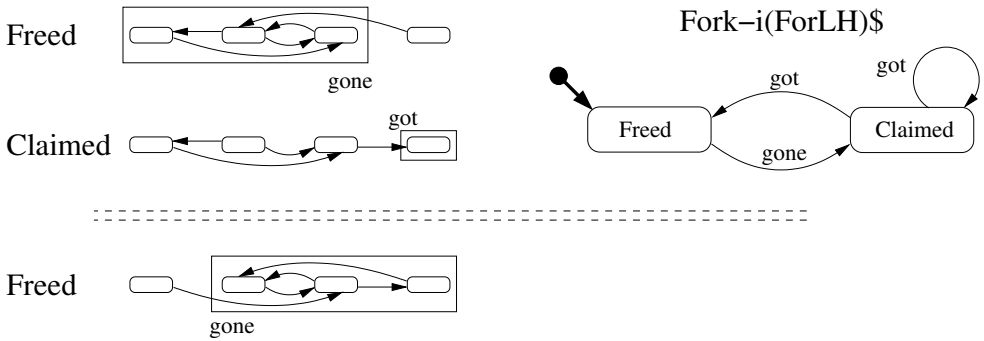


wrt $Phil_{i-1}$'s right hand: constraints on $Fork_i$

- arrive at its port-a
- are: phase *Freed*

animation of Dining Philosophers

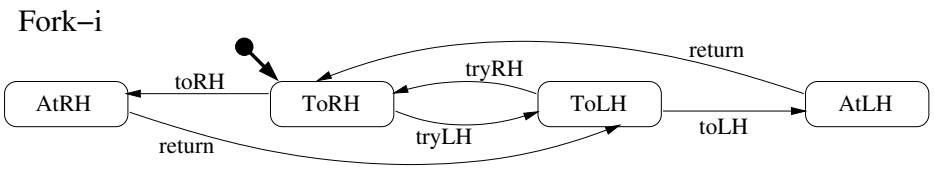


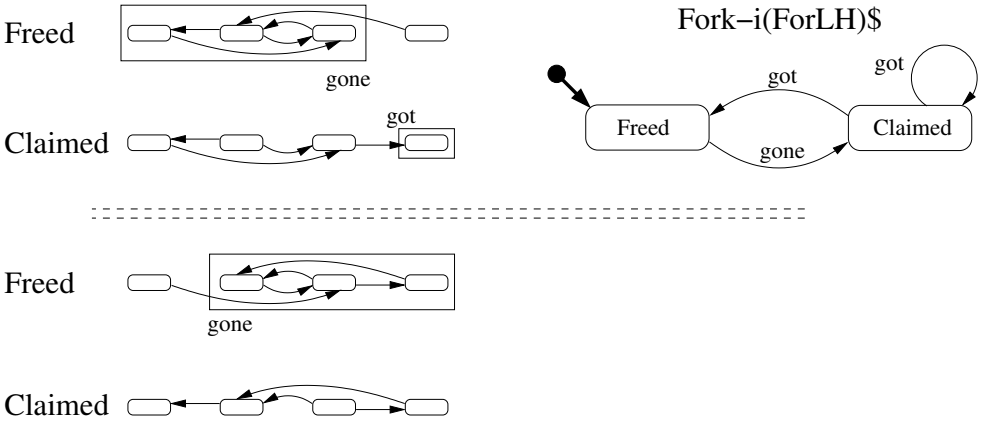


wrt $Phil_{i-1}$'s right hand: constraints on $Fork_i$

- arrive at its port-a
- are: phase *Freed* ; trap *gone*

animation of Dining Philosophers

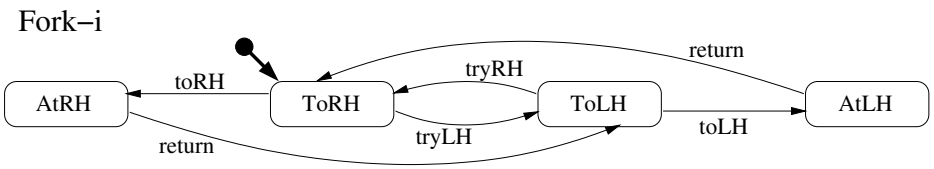


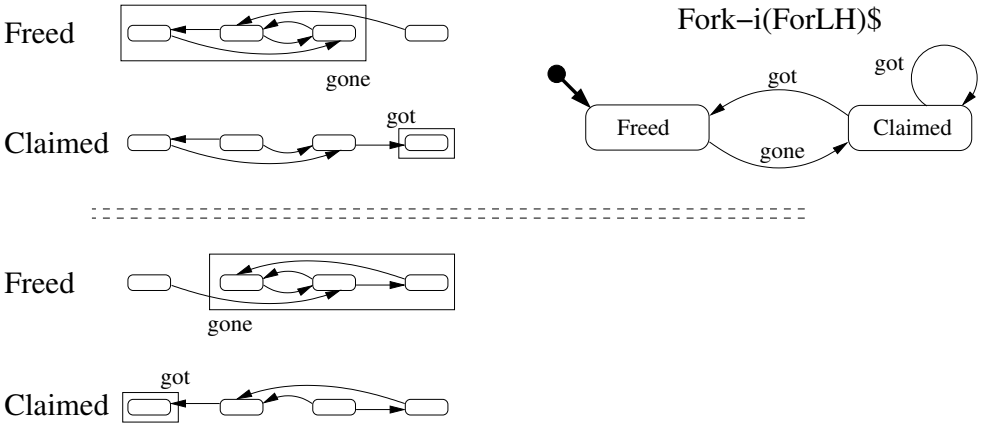


wrt $Phil_{i-1}$'s right hand: constraints on $Fork_i$

- arrive at its port-a
- are: phases *Freed*, *Claimed*; trap *gone*

animation of Dining Philosophers

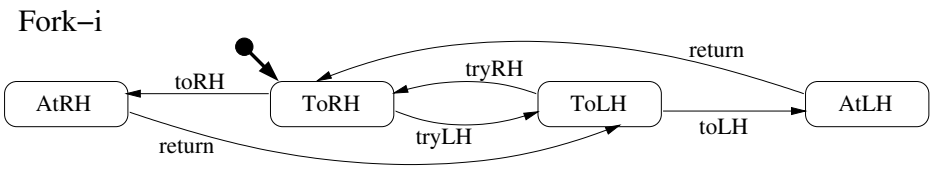


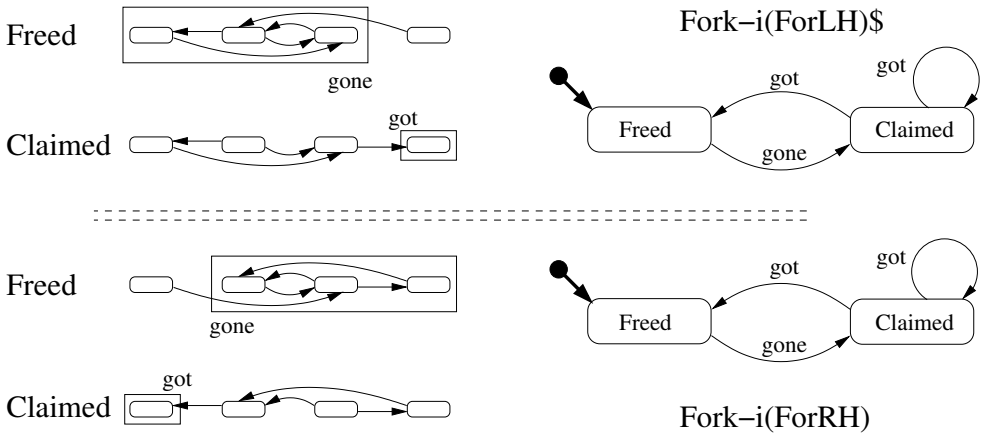


wrt $Phil_{i-1}$'s right hand: constraints on $Fork_i$

- arrive at its port-a
- are: phases *Freed*, *Claimed*; traps *gone*, *got*

animation of Dining Philosophers

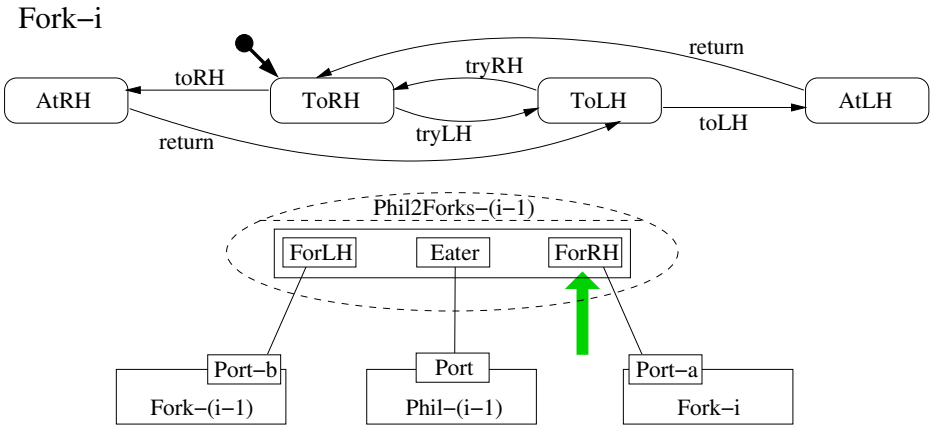


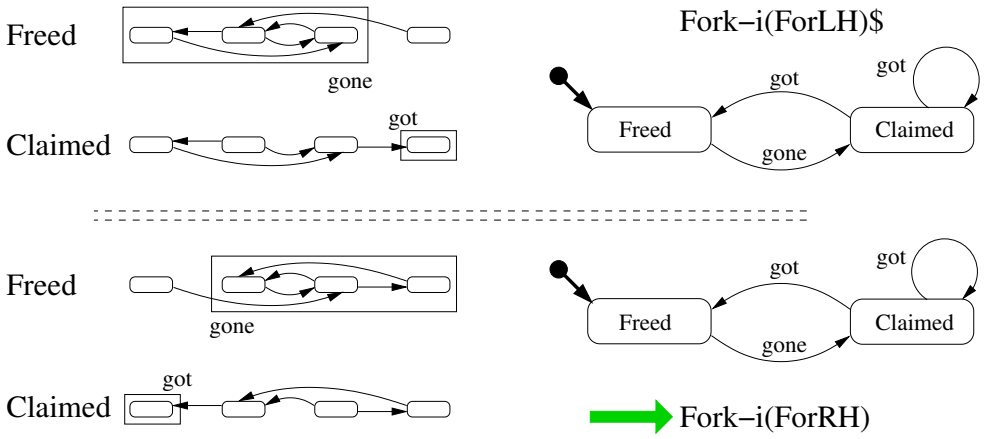


wrt $Phil_{i-1}$'s right hand: constraints on $Fork_i$

- arrive at its port-a
- are: phases *Freed*, *Claimed*; traps *gone*, *got*
- and via role $Fork_i(ForRH)$

animation of Dining Philosophers





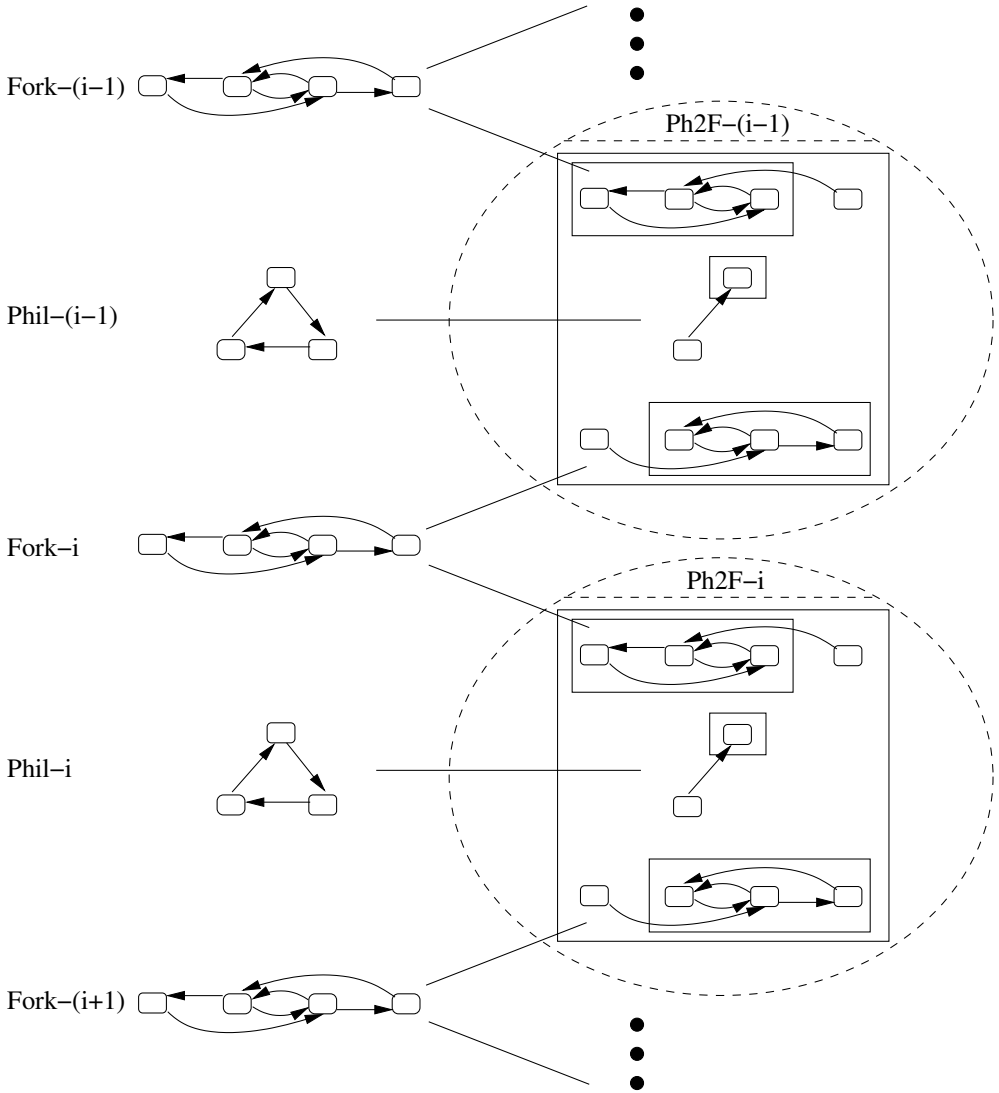
wrt $Phil_{i-1}$'s right hand: constraints on $Fork_i$

- arrive at its port-a
- are: phases *Freed*, *Claimed*; traps *gone*, *got*
- and are governed via role $Fork_i(ForRH)$

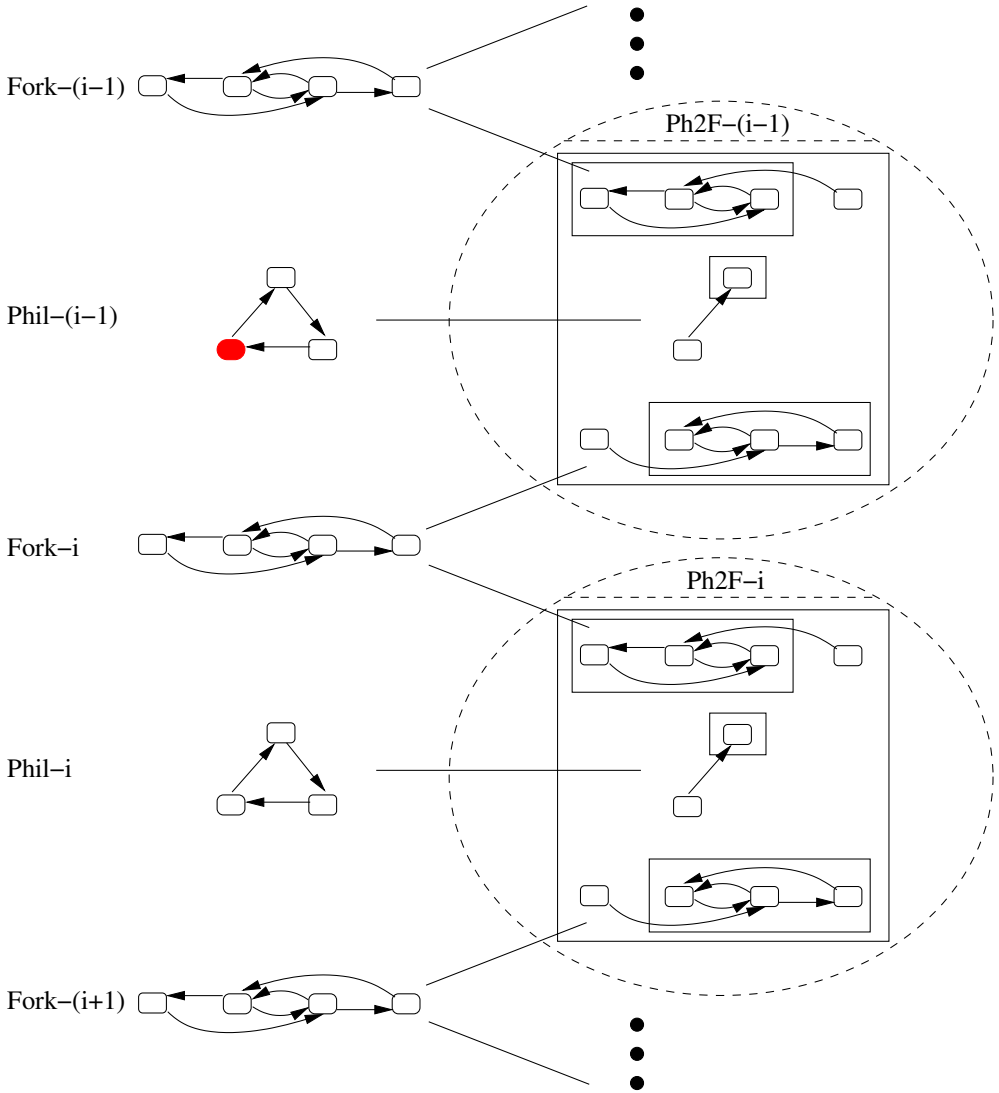
animation of Dining Philosophers

Where to start?

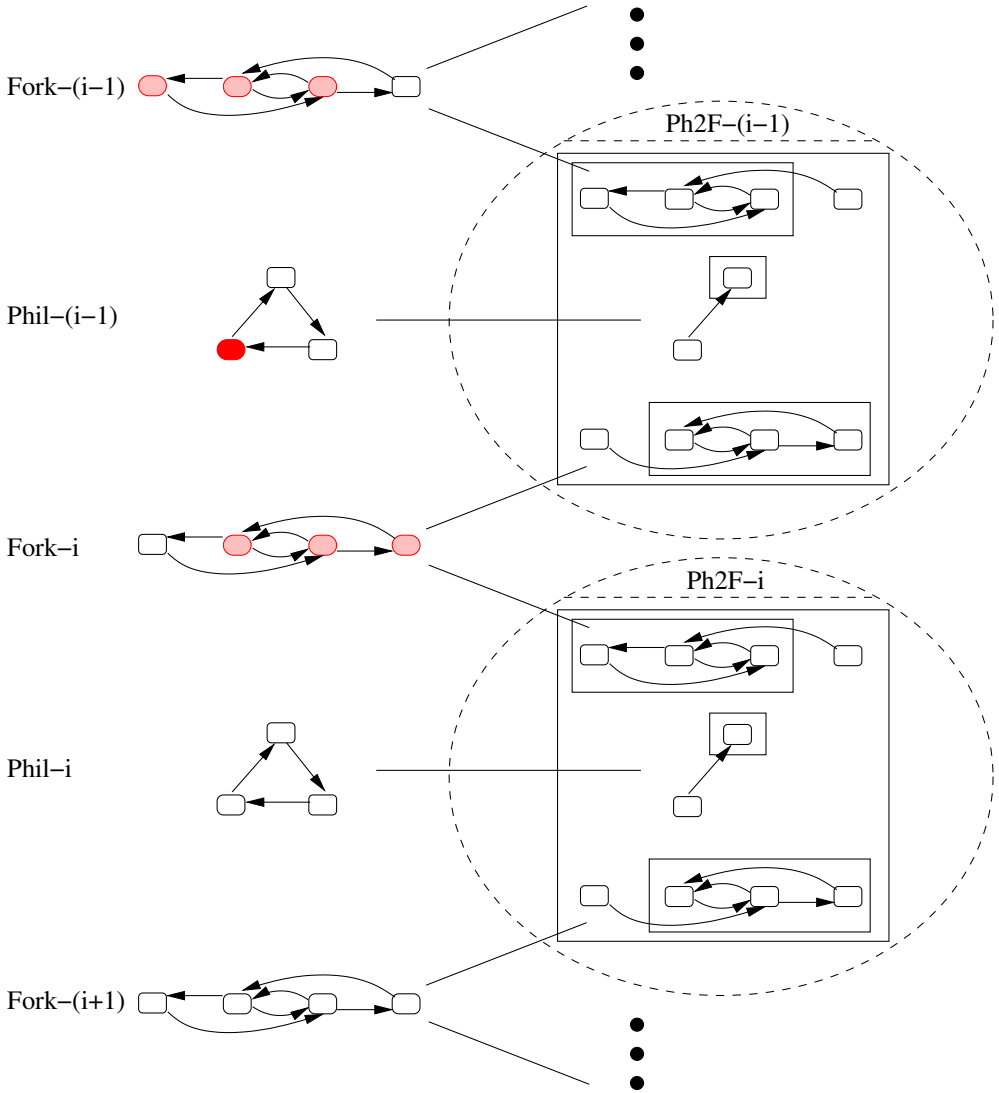
animation of as-is coordination



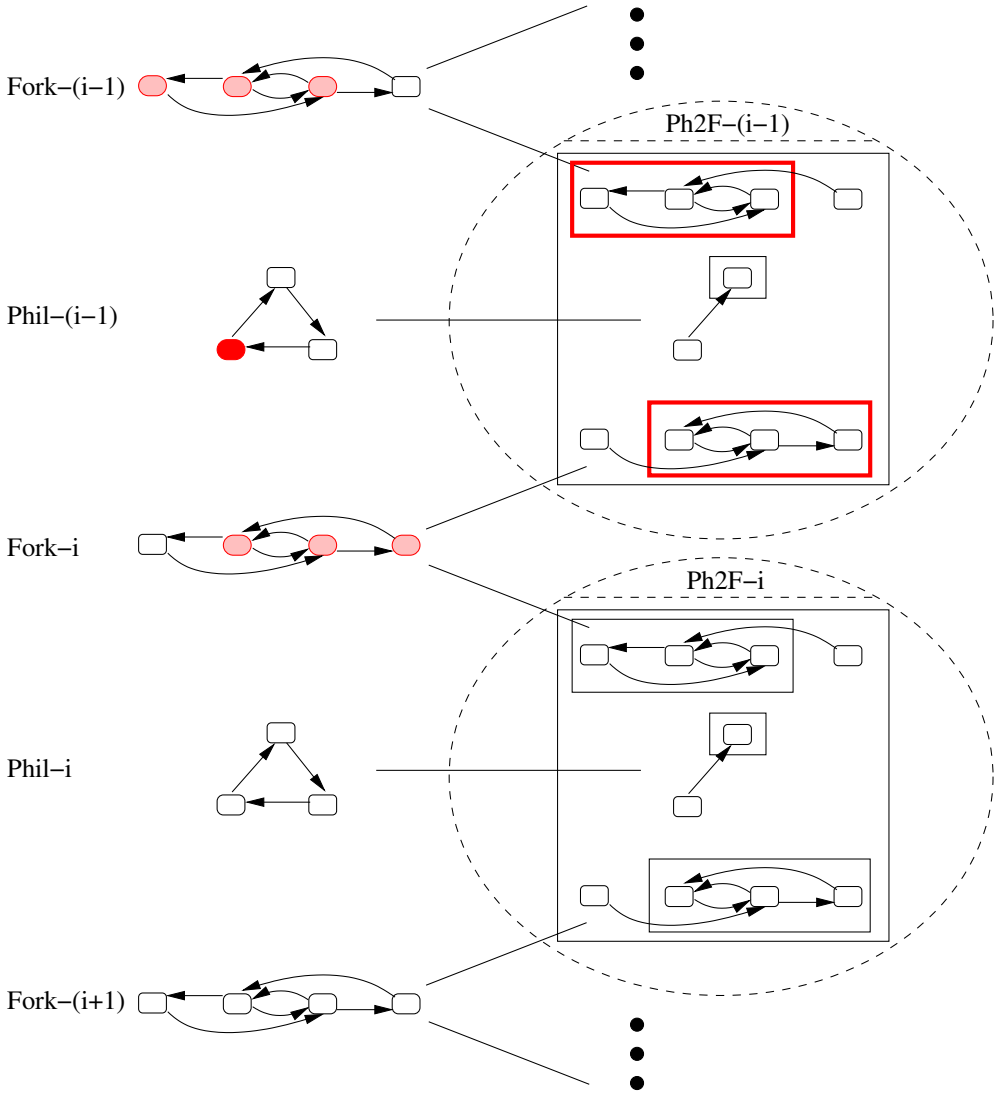
animation of as-is coordination



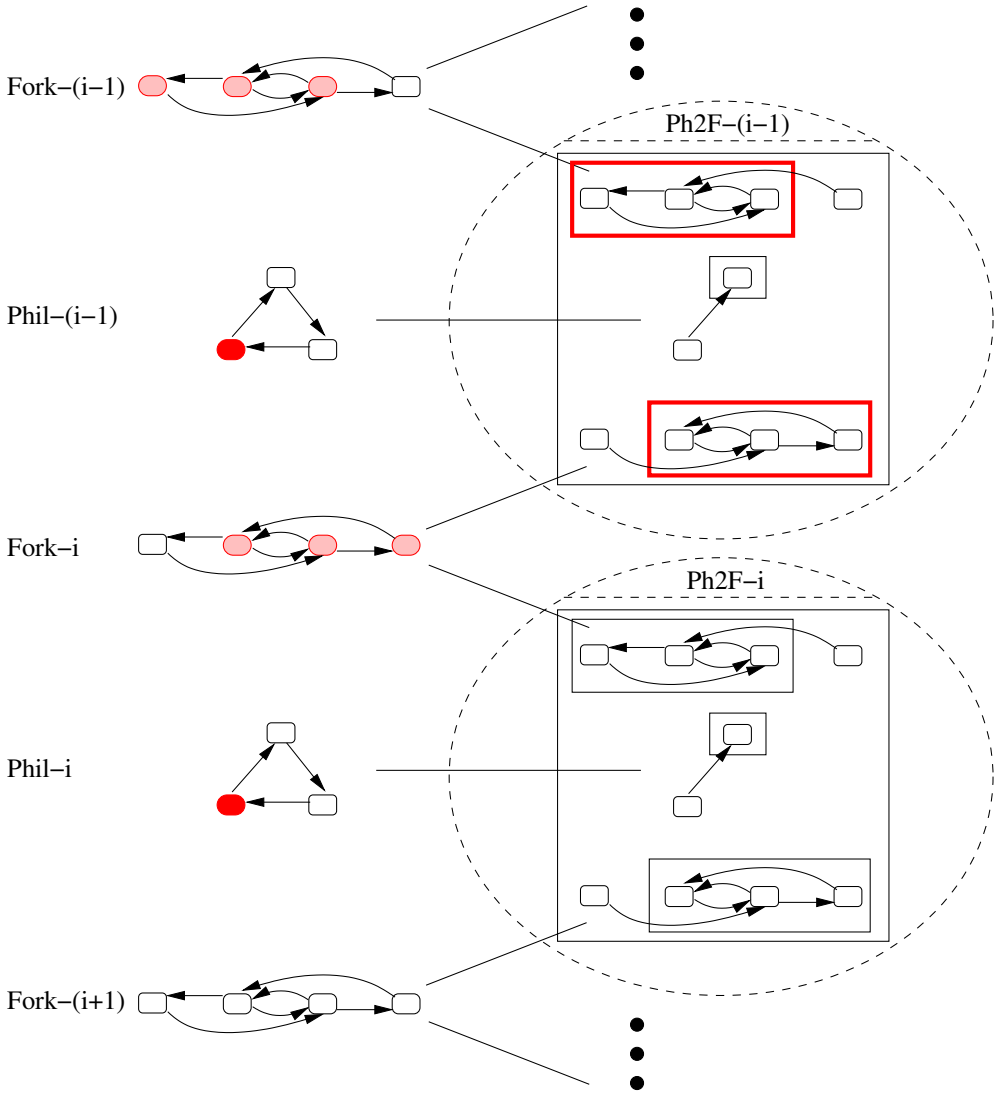
animation of as-is coordination



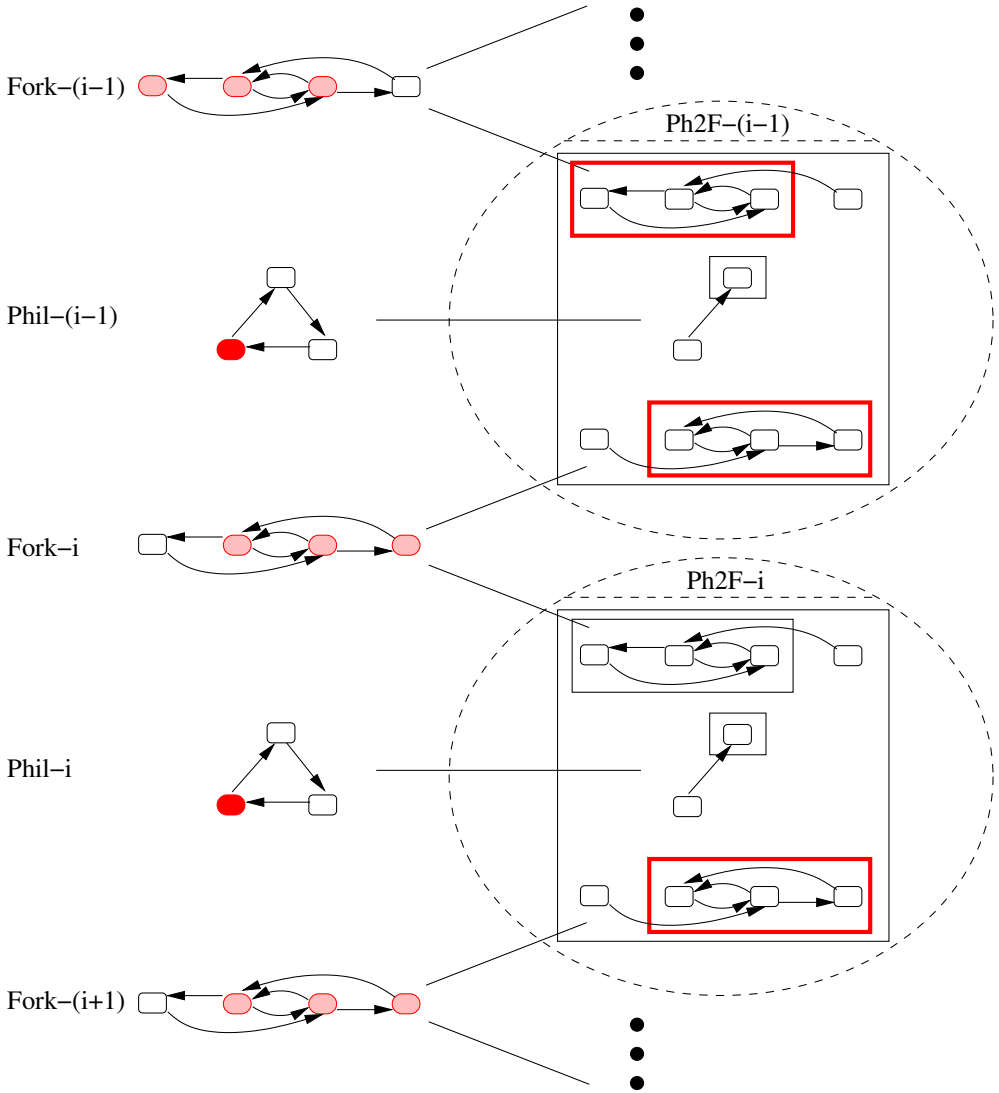
animation of as-is coordination



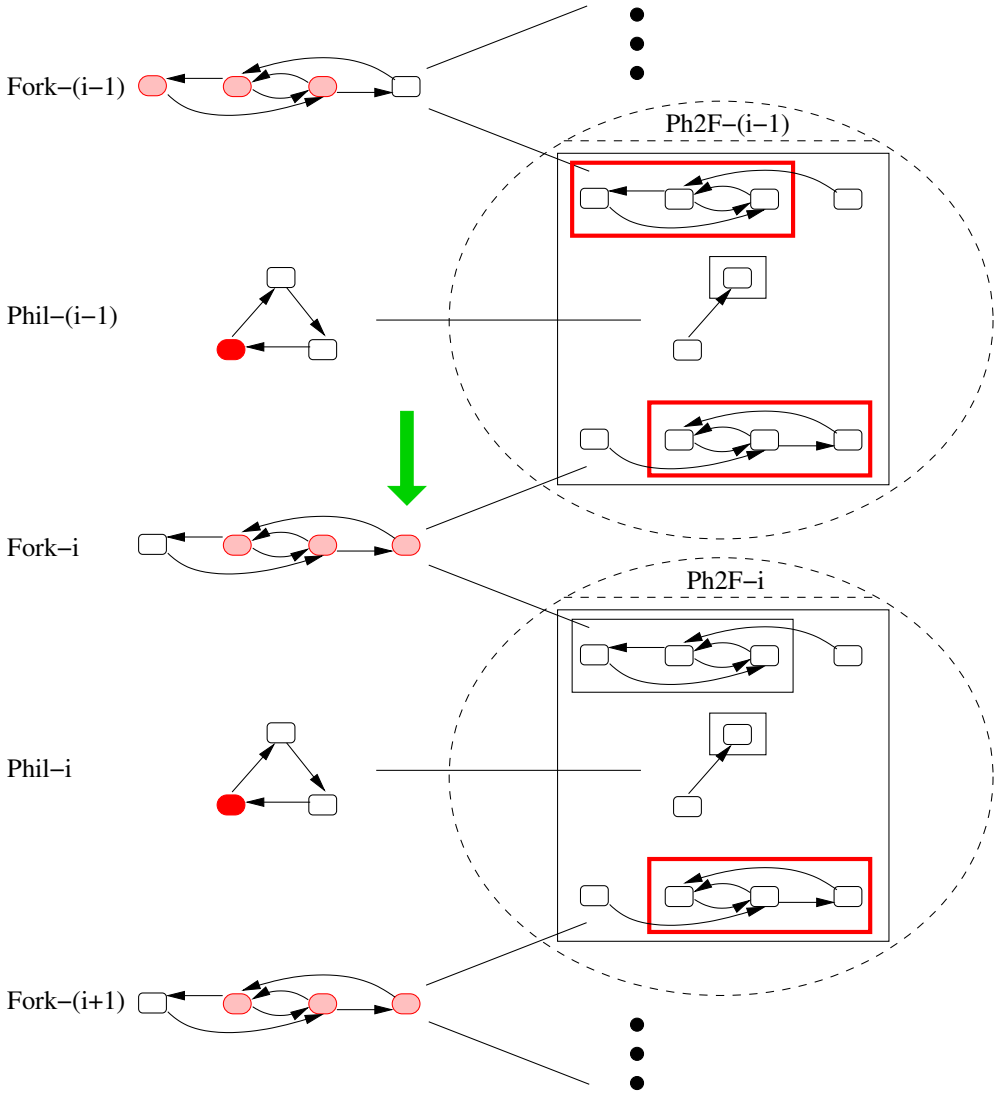
animation of as-is coordination



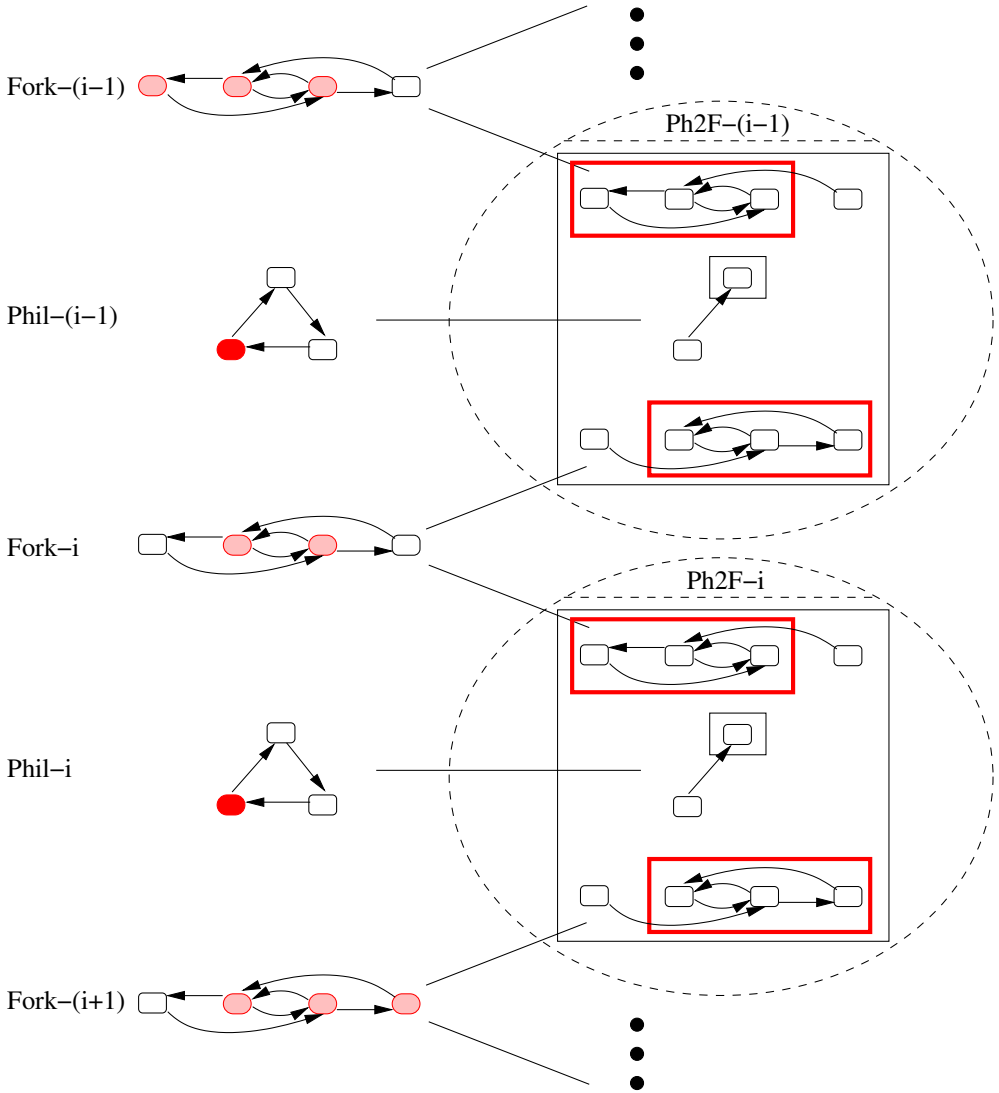
animation of as-is coordination



animation of as-is coordination



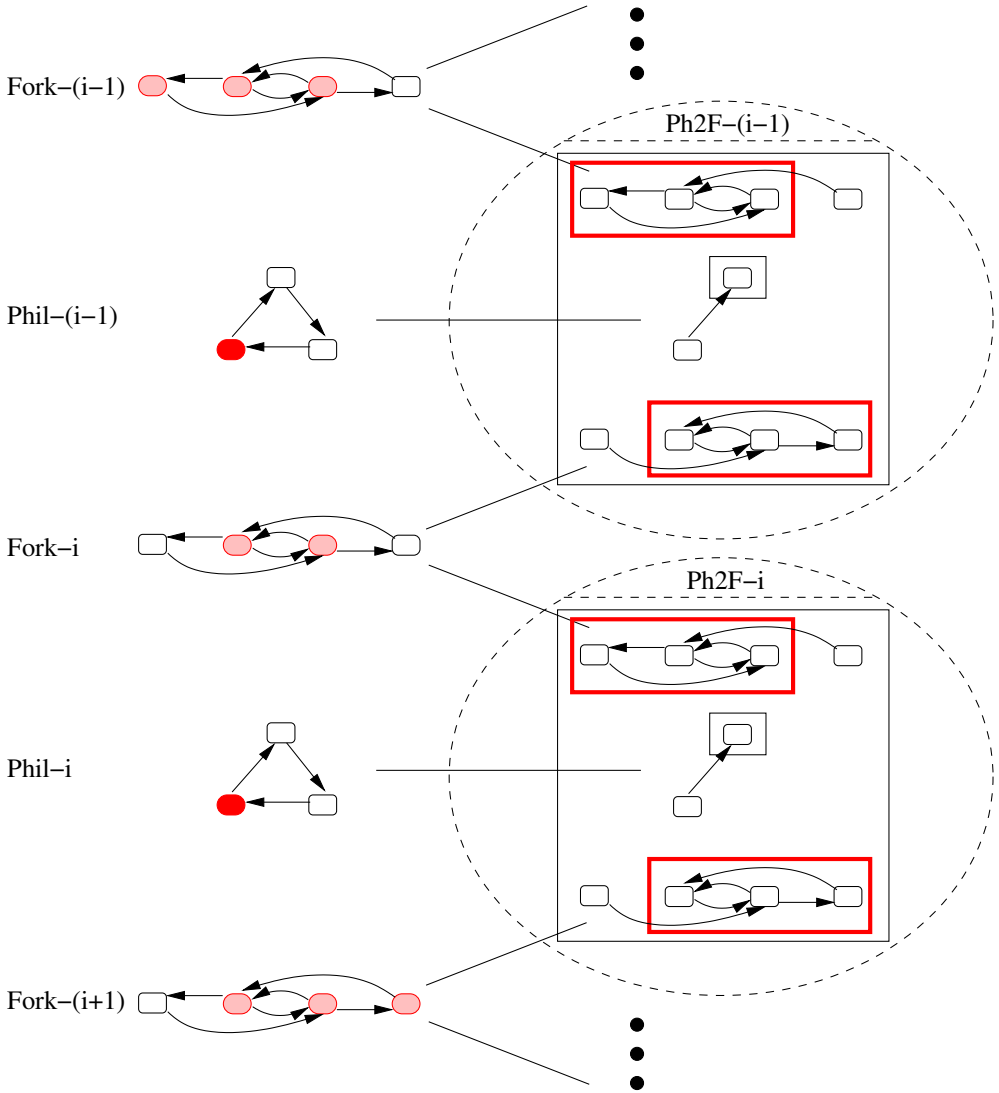
animation of as-is coordination



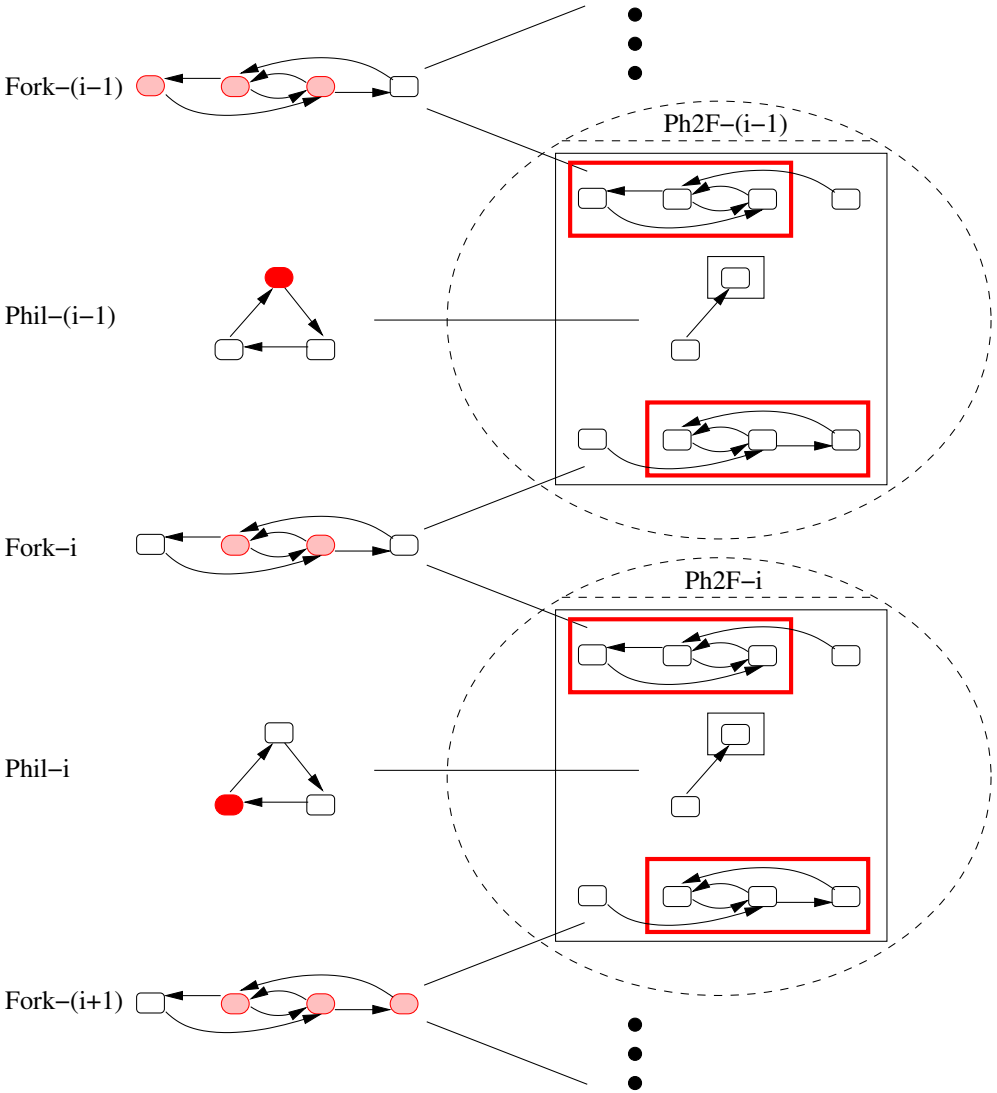
animation of as-is coordination

How to move on?

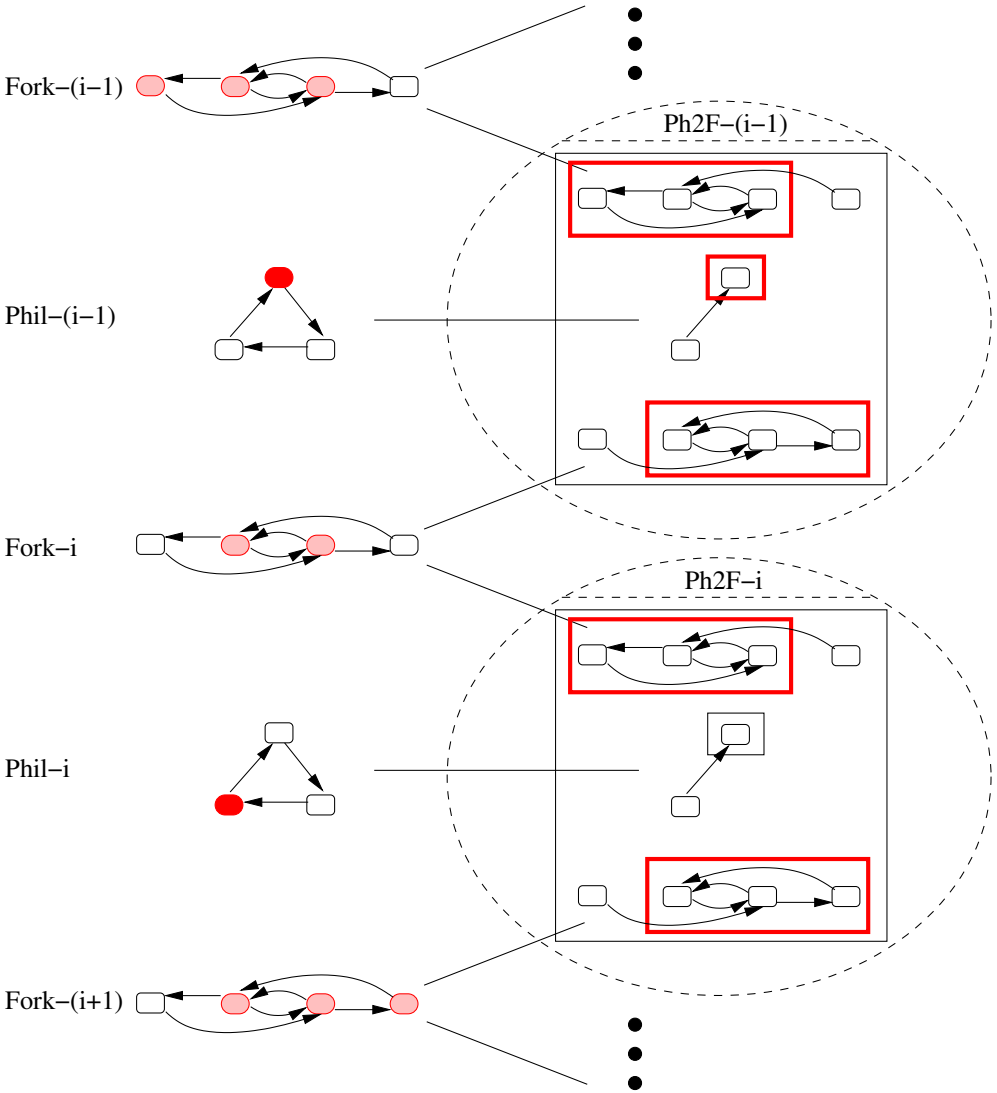
animation of as-is coordination



animation of as-is coordination



animation of as-is coordination



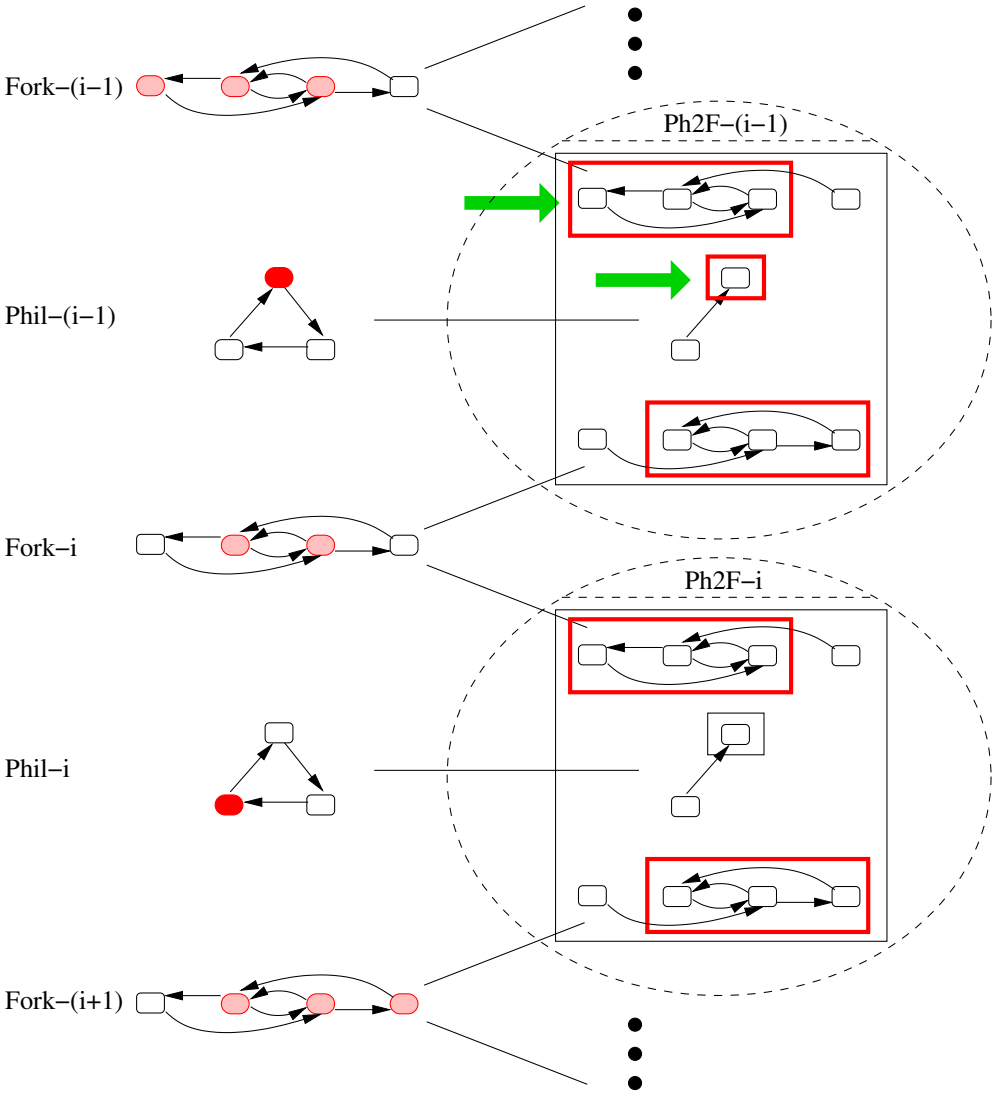
animation of as-is coordination

consistency rule

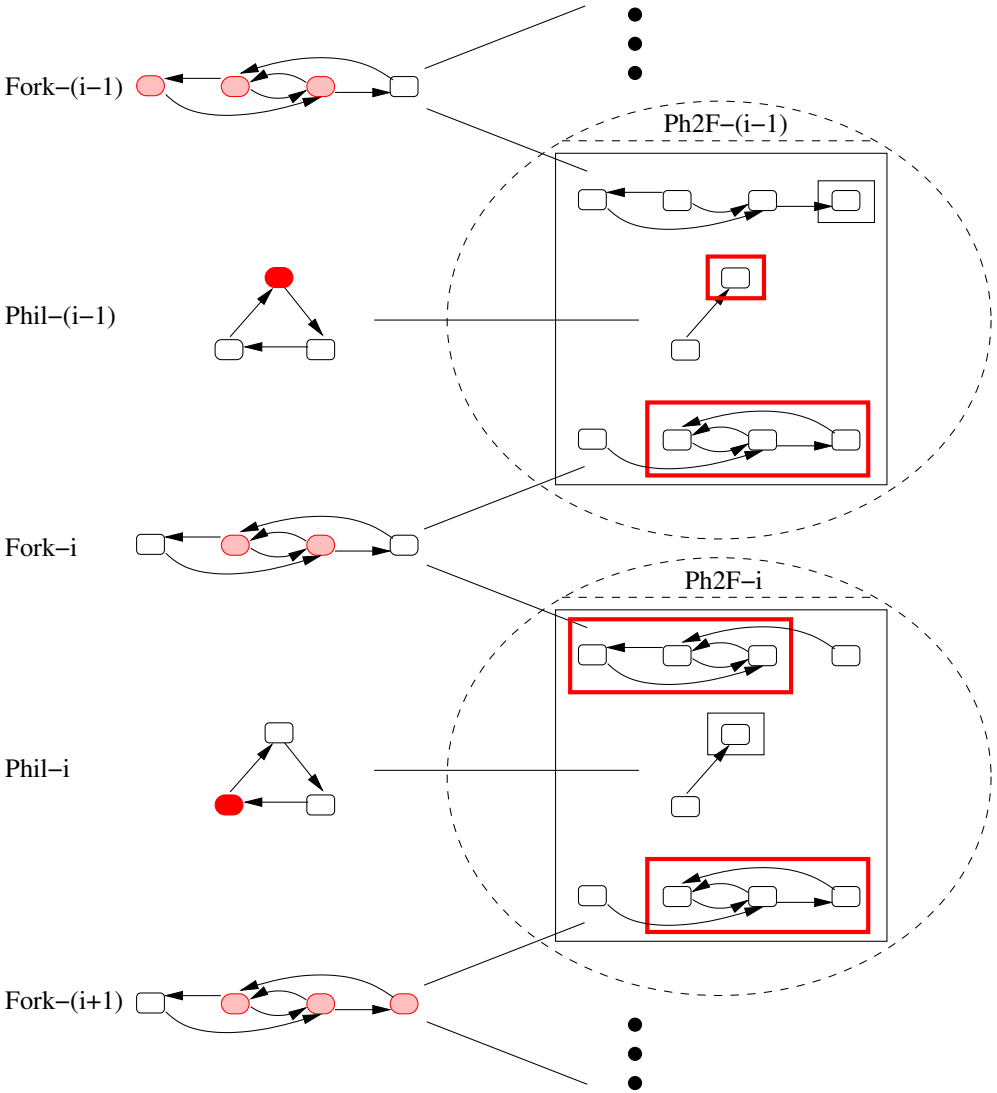
* $Phil_{i-1}(Eater)\colon Disallowed \xrightarrow{\text{request}} Disallowed$,
  $Fork_{i-1}(ForLH)\colon Freed \xrightarrow{\text{gone}} Claimed$

is now applicable

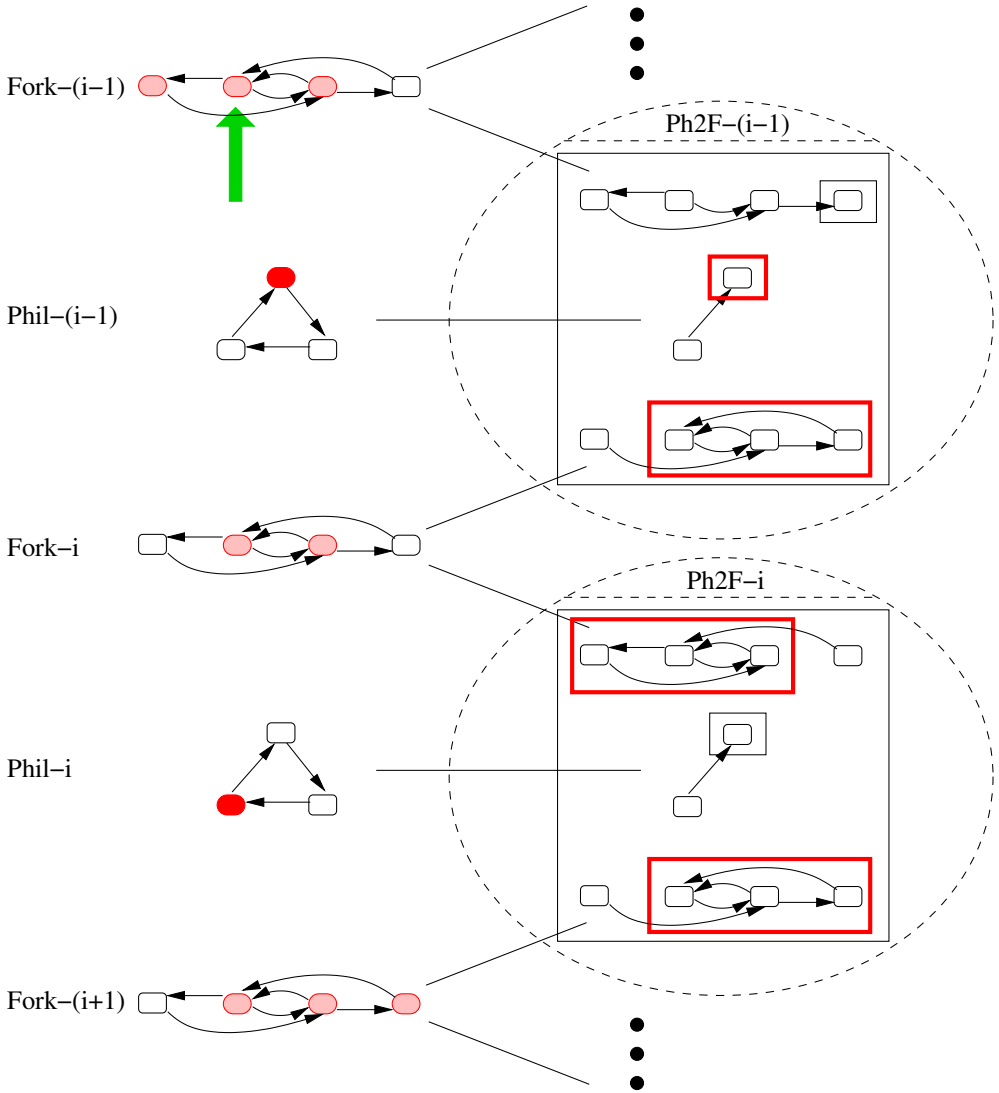
animation of as-is coordination



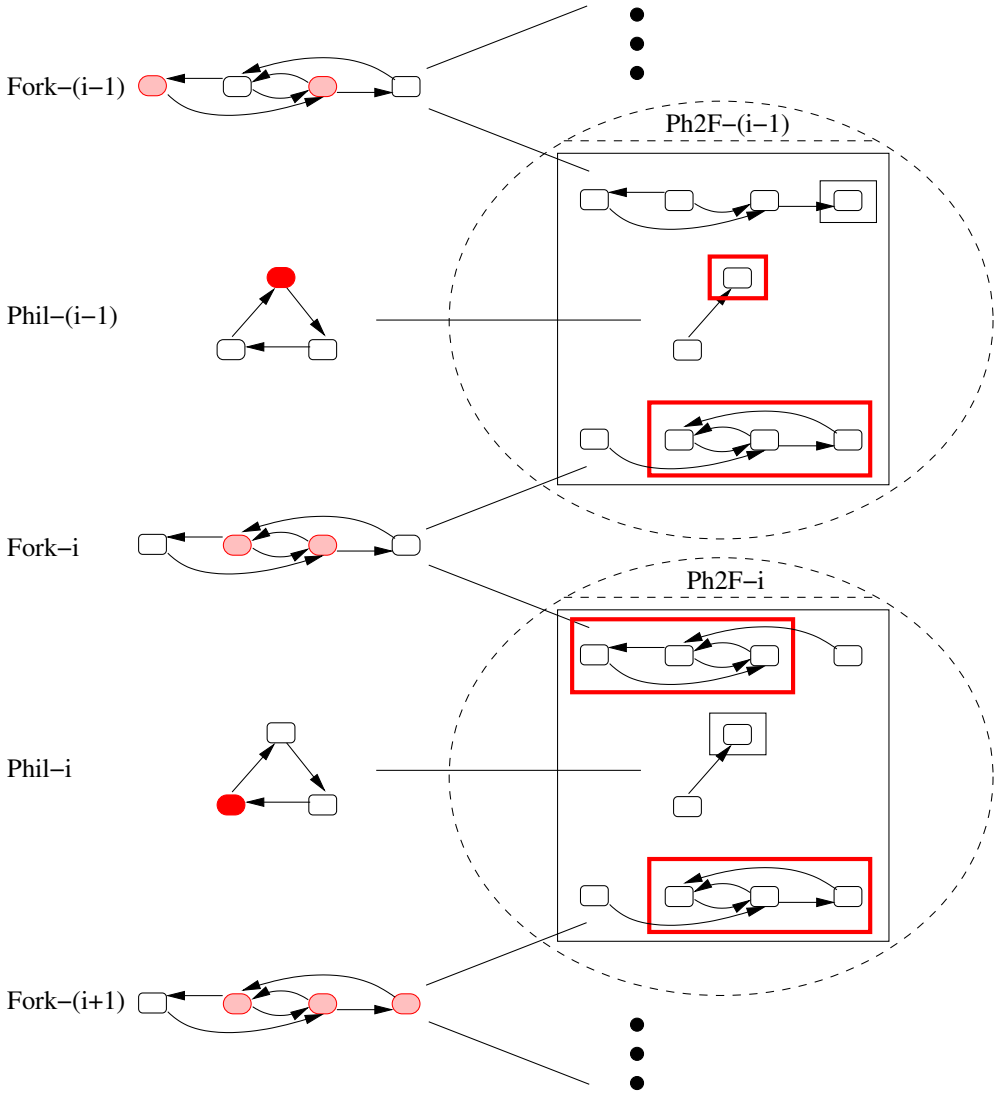
animation of as-is coordination



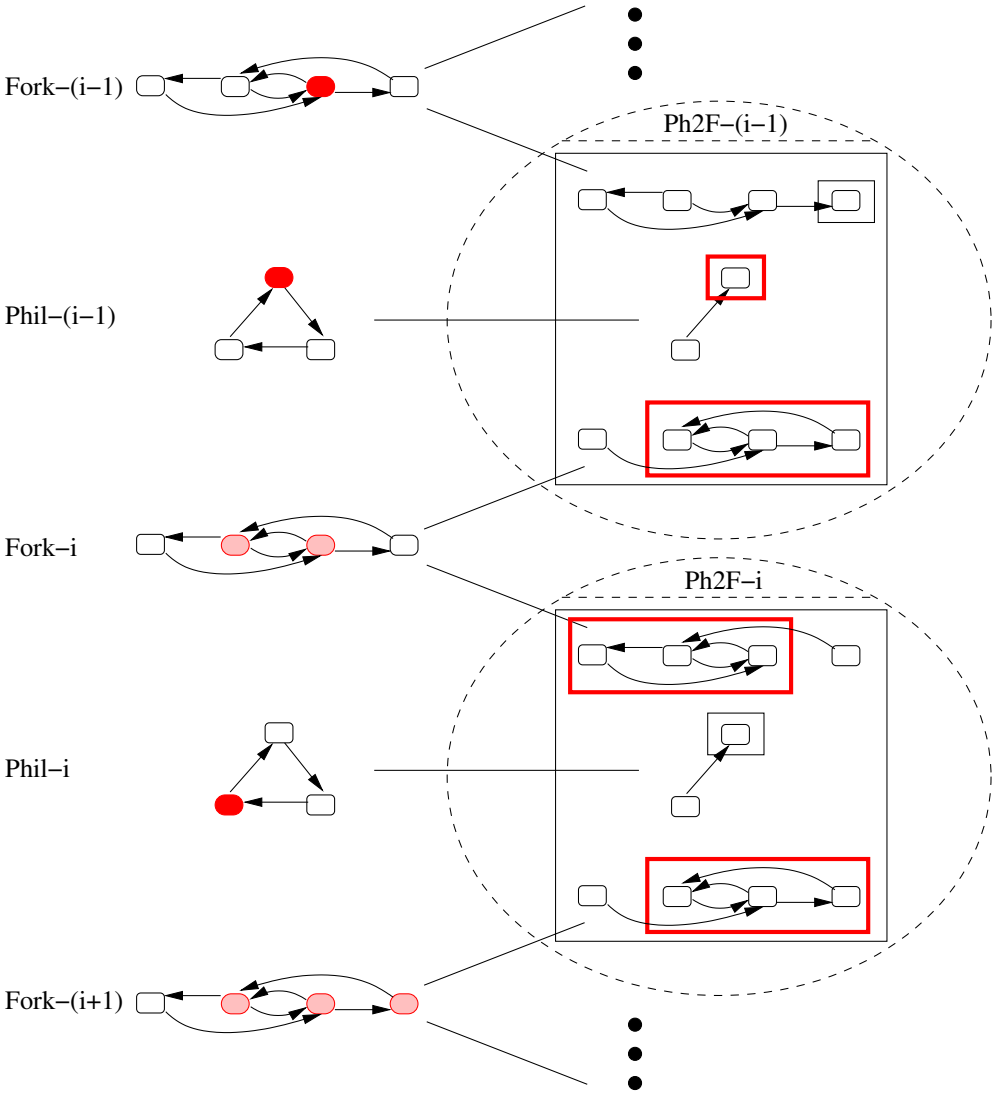
animation of as-is coordination



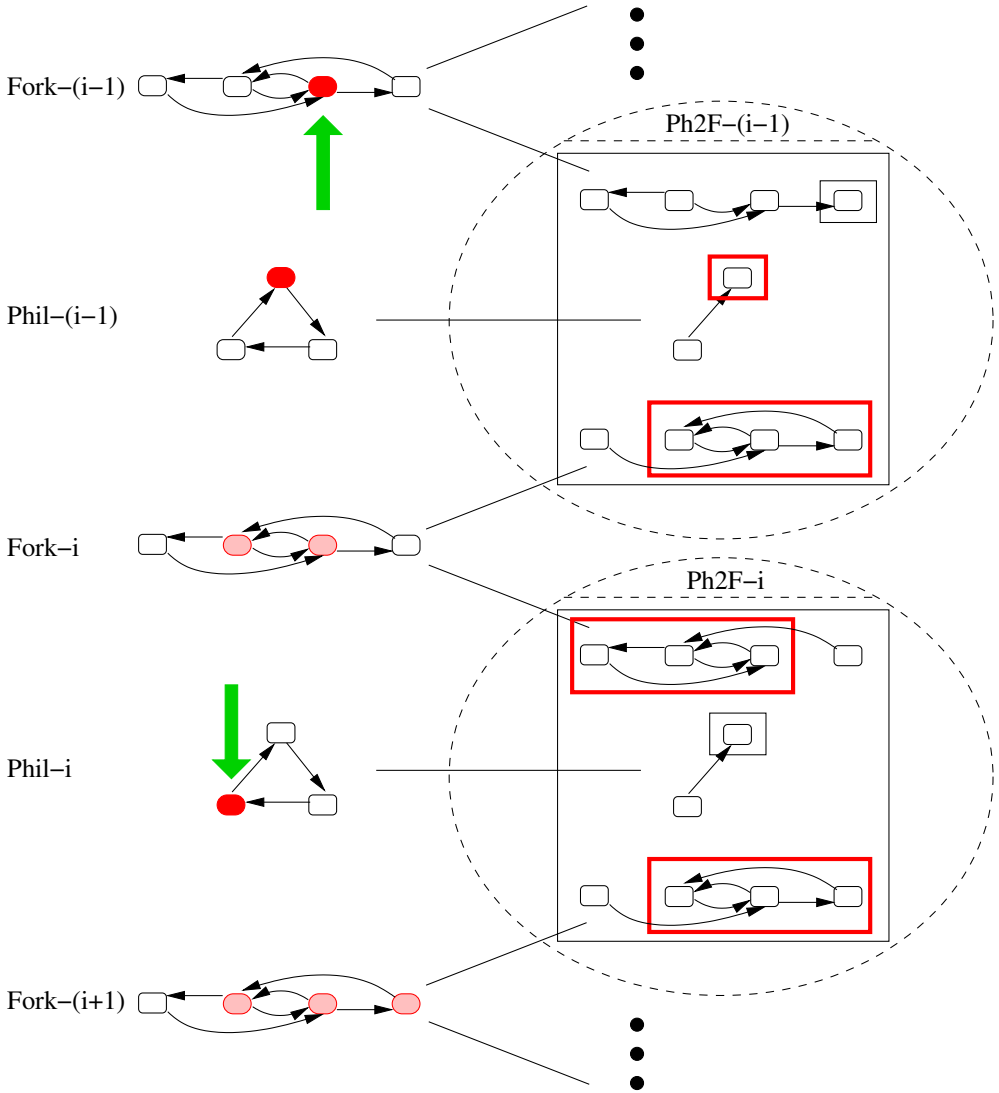
animation of as-is coordination



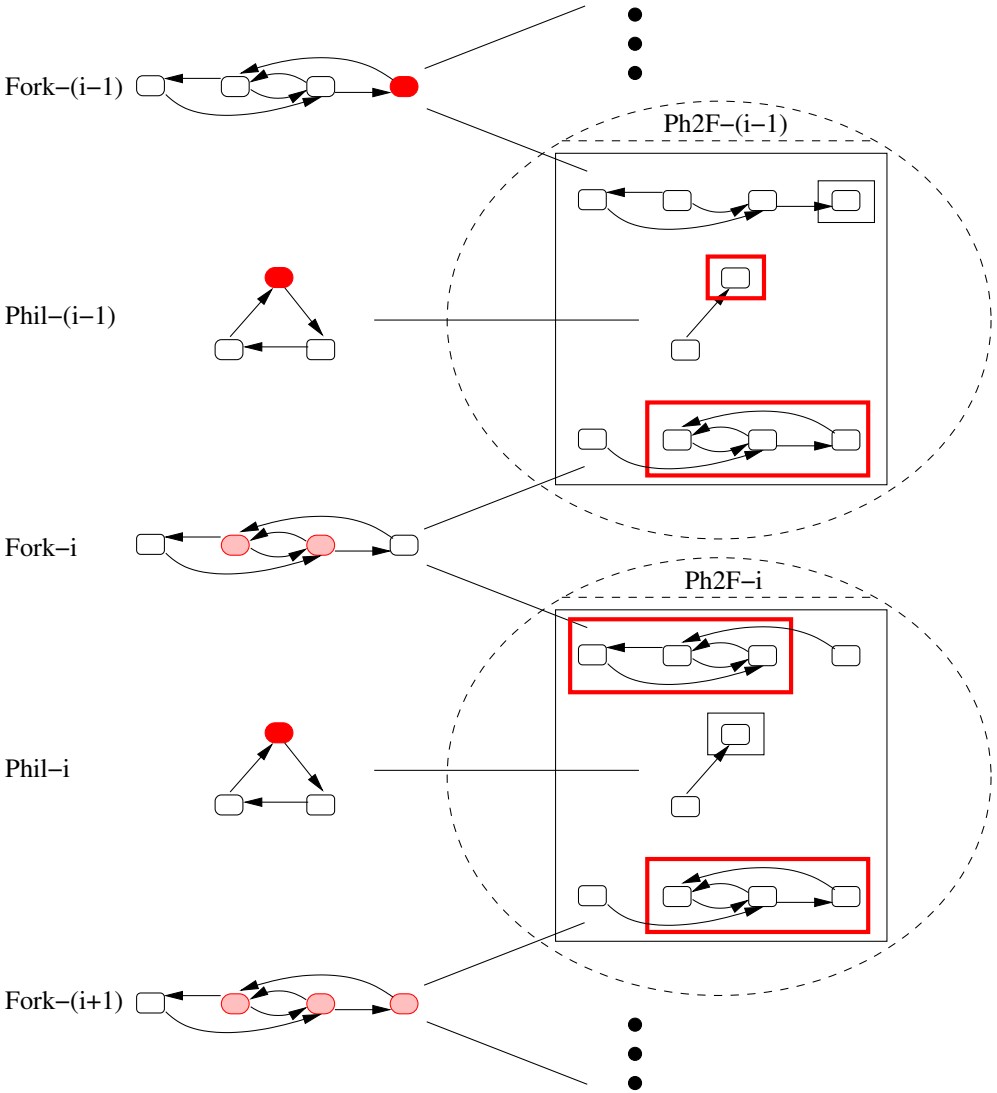
animation of as-is coordination



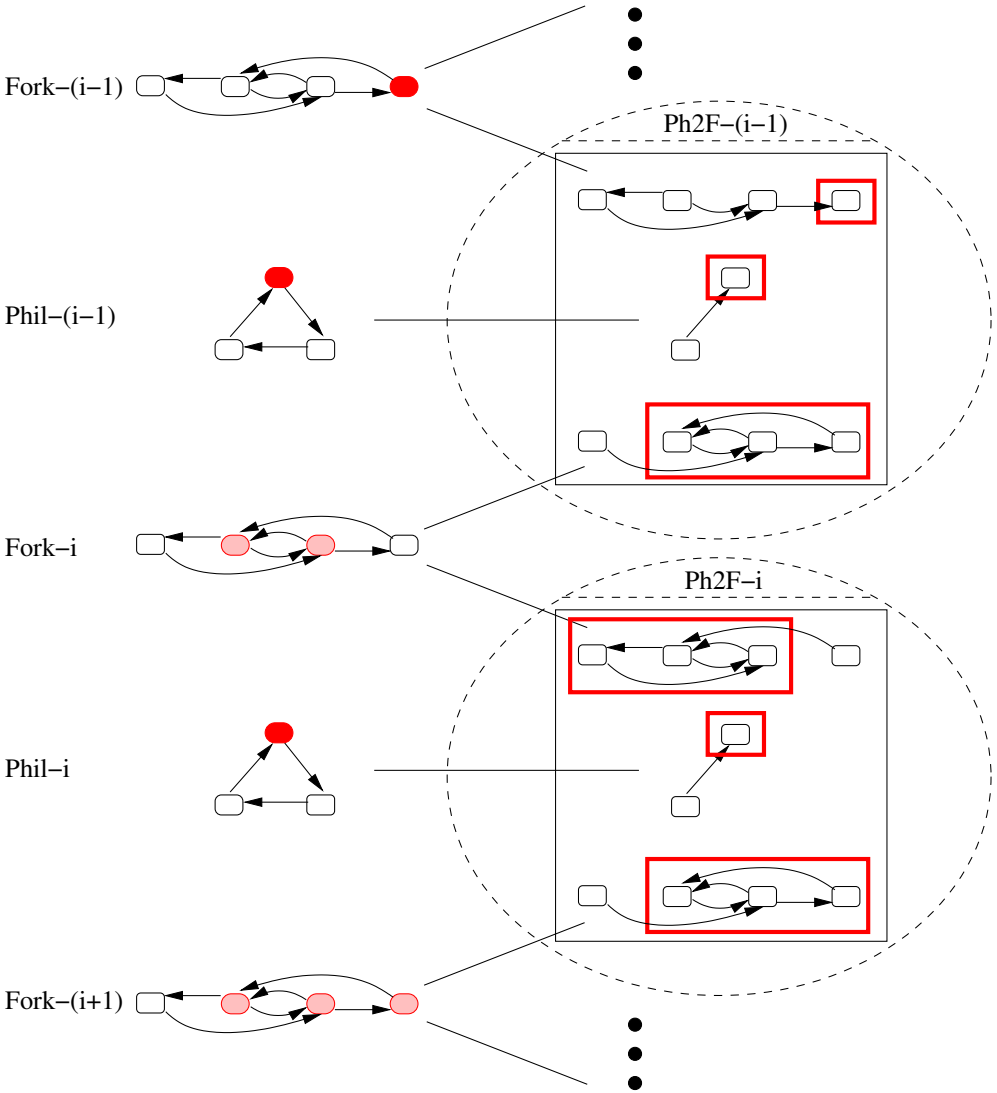
animation of as-is coordination



animation of as-is coordination



animation of as-is coordination



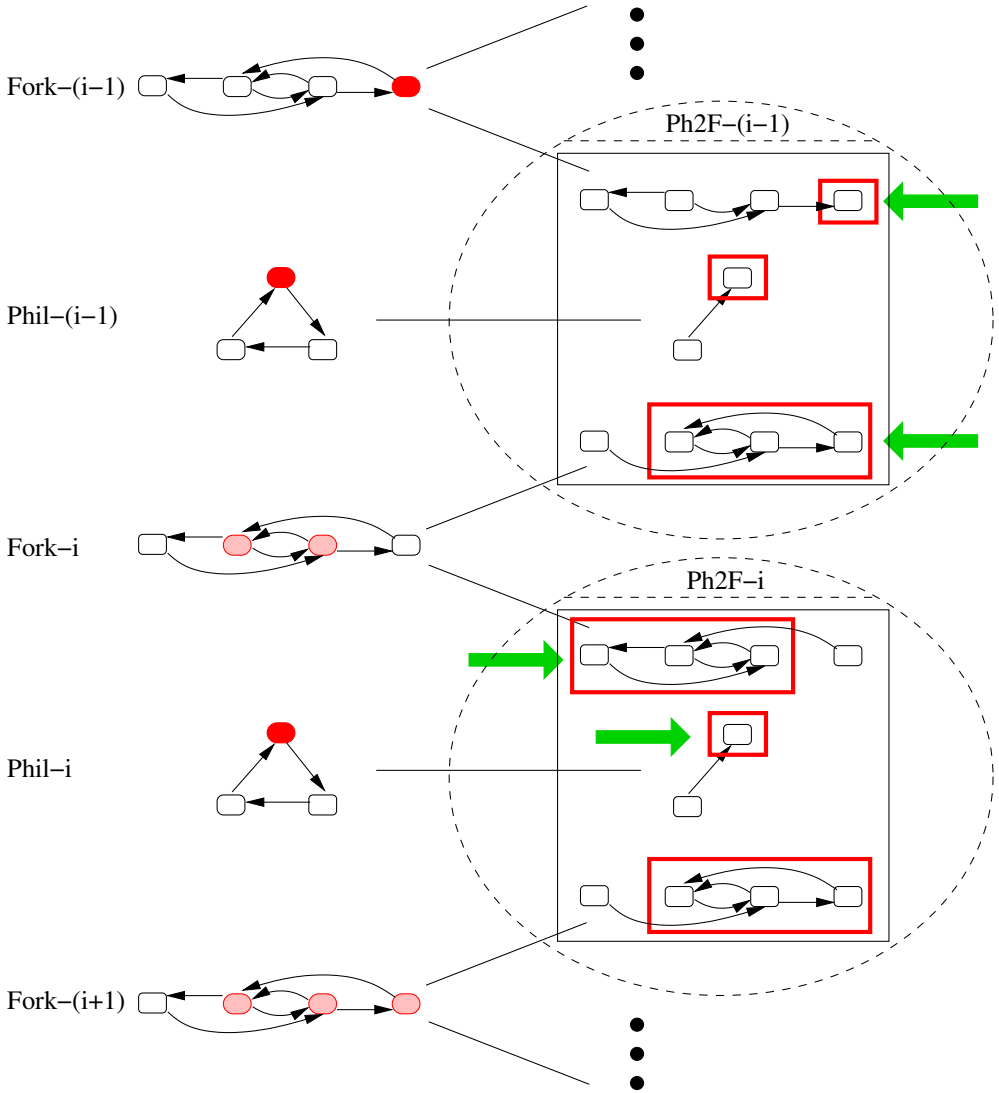
animation of as-is coordination

two consistency rules

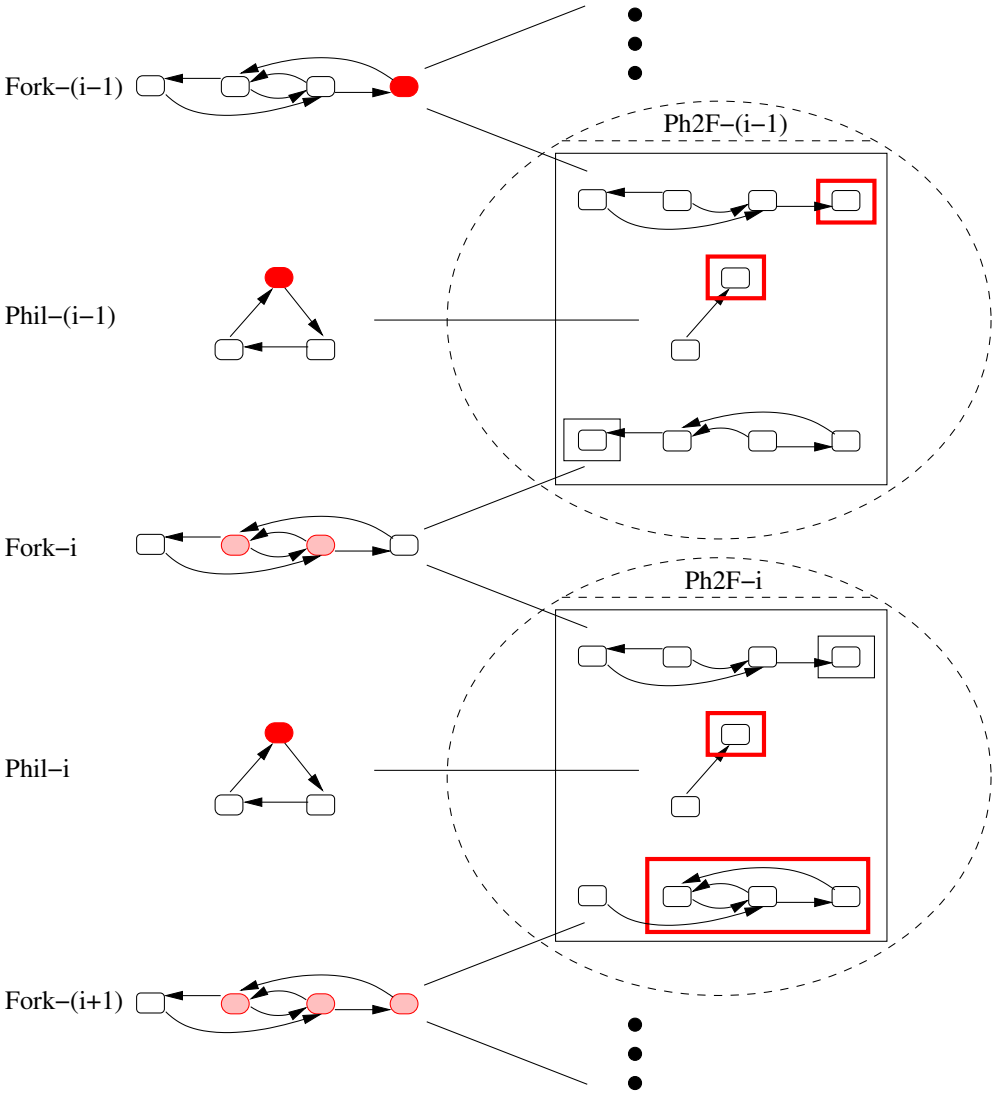
* $Fork_{i-1}(ForLH)$: $Claimed \xrightarrow{\text{got}} Claimed$,
  $Fork_i(ForRH)$: $Freed \xrightarrow{\text{gone}} Claimed$
* $Phil_i(Eater)$: $Disallowed \xrightarrow{\text{request}} Disallowed$,
  $Fork_i(ForLH)$: $Freed \xrightarrow{\text{gone}} Claimed$

are applicable, simultaneously

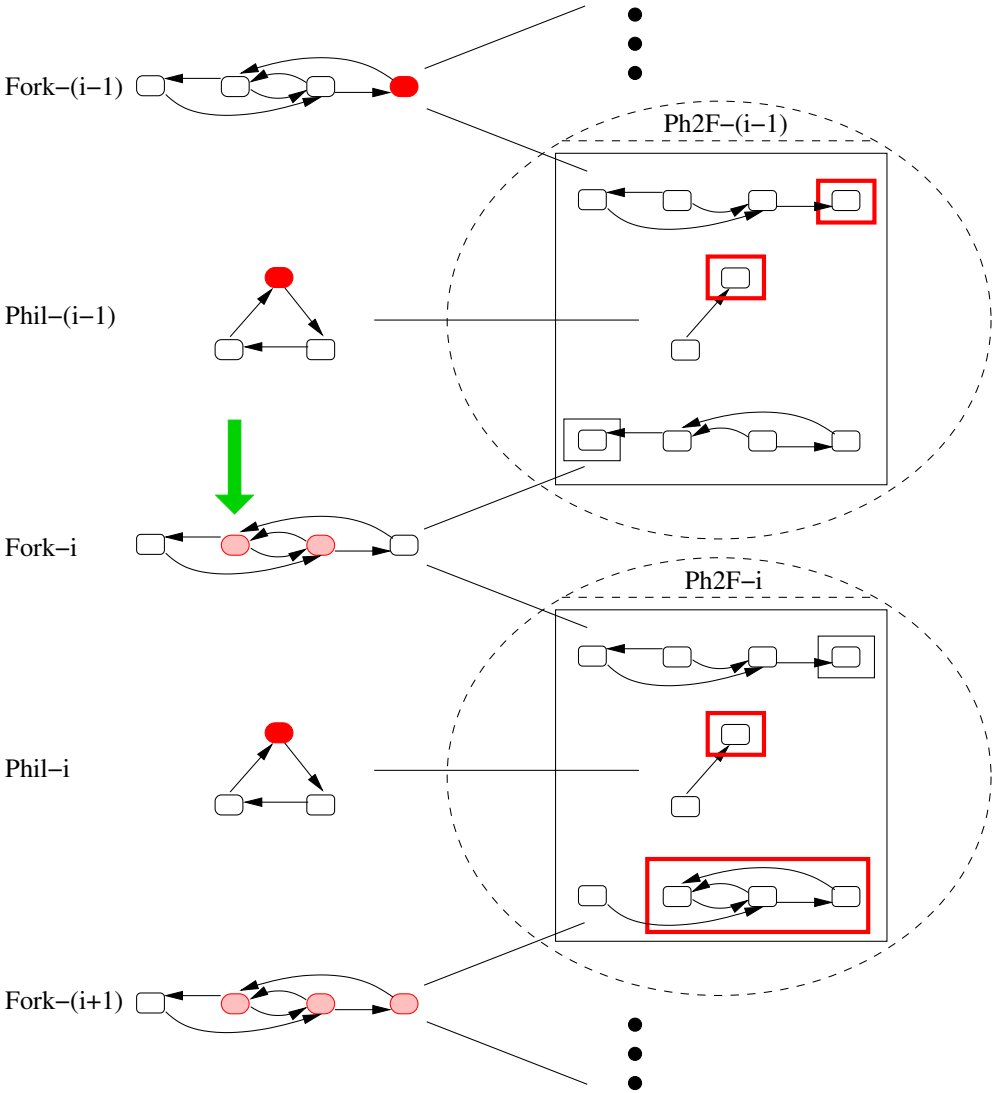
animation of as-is coordination



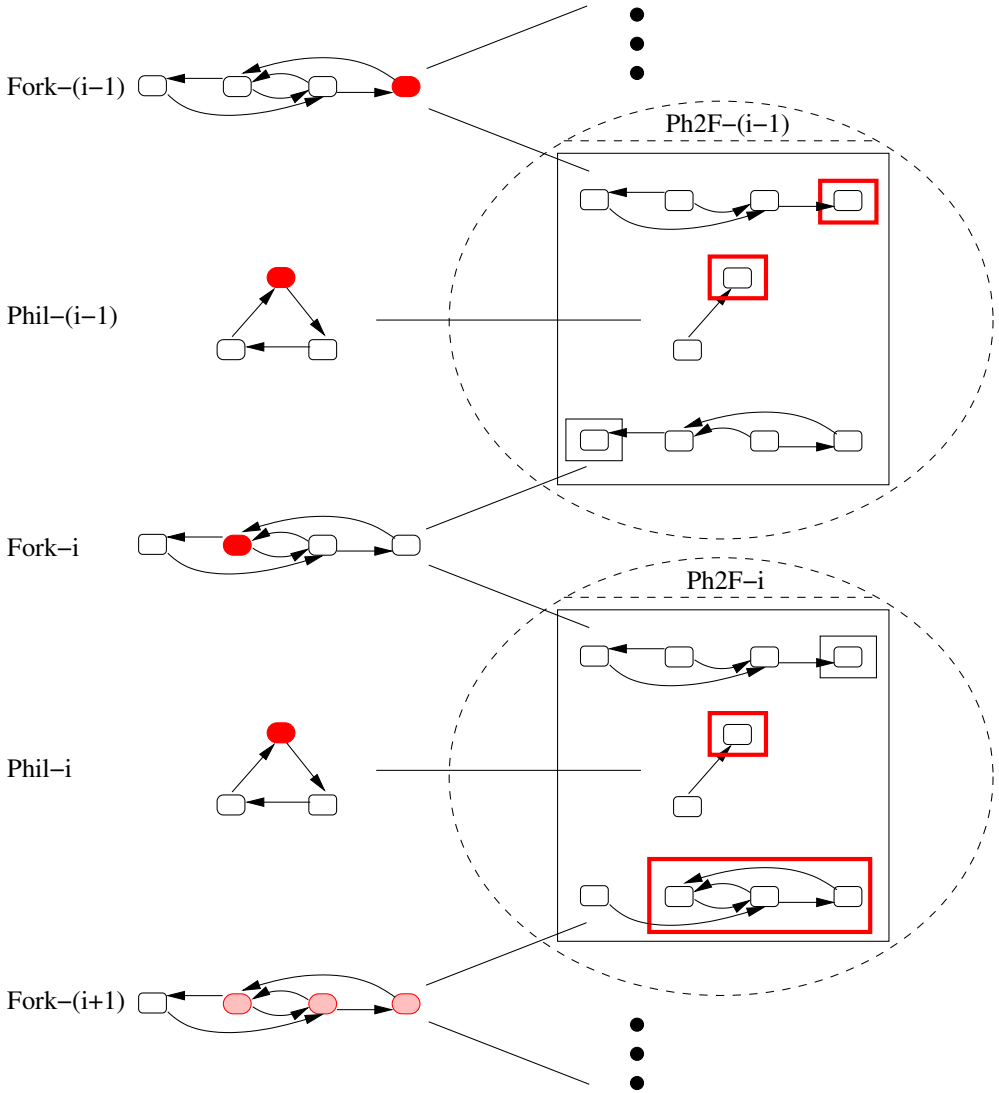
animation of as-is coordination



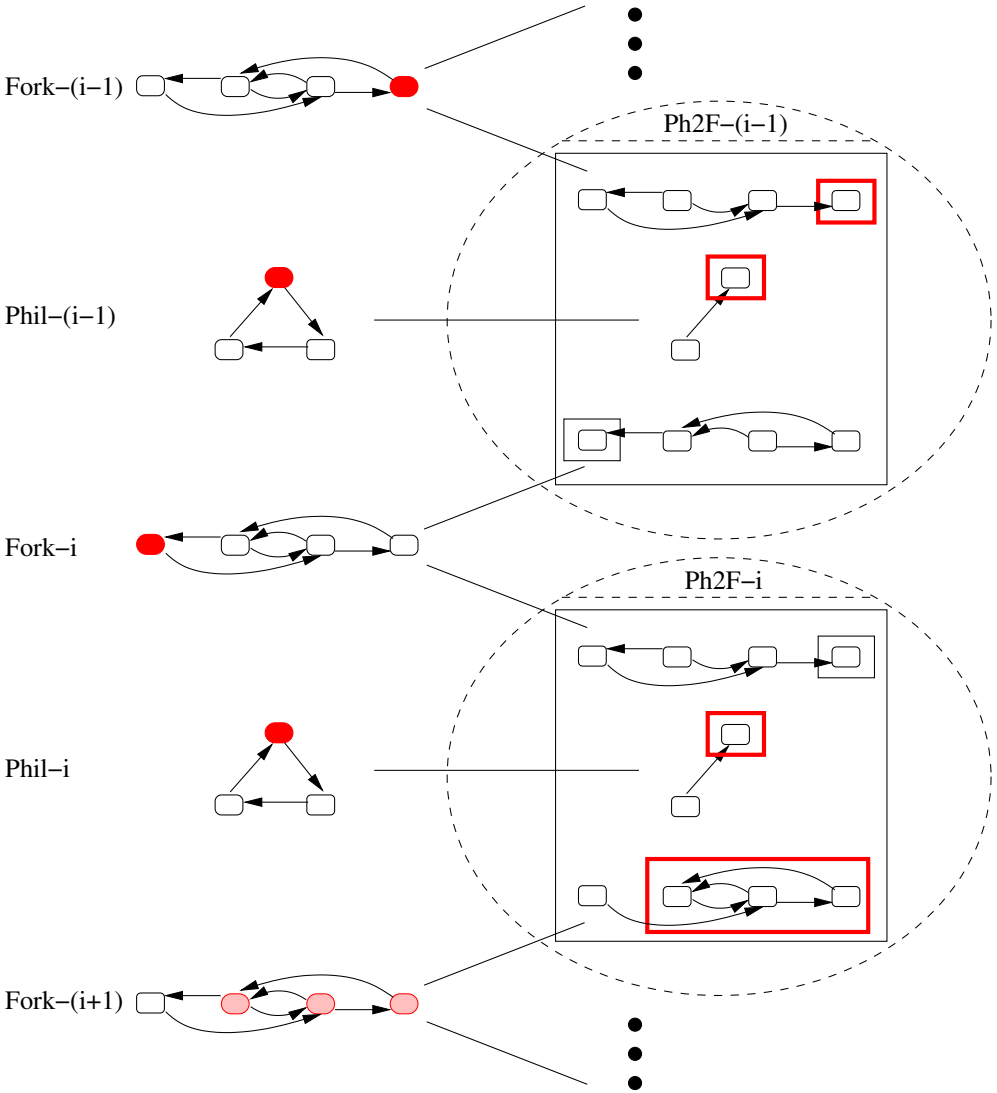
animation of as-is coordination



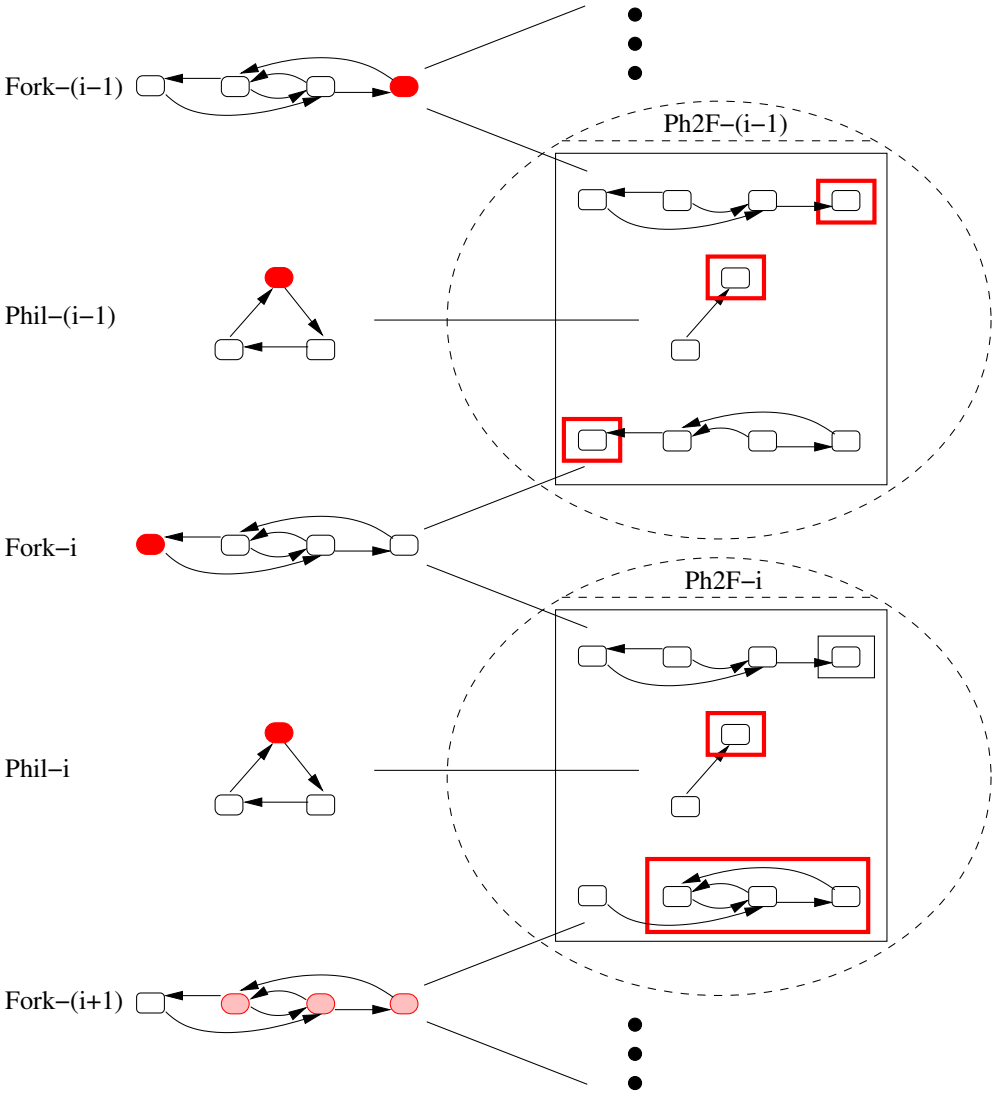
animation of as-is coordination



animation of as-is coordination



animation of as-is coordination



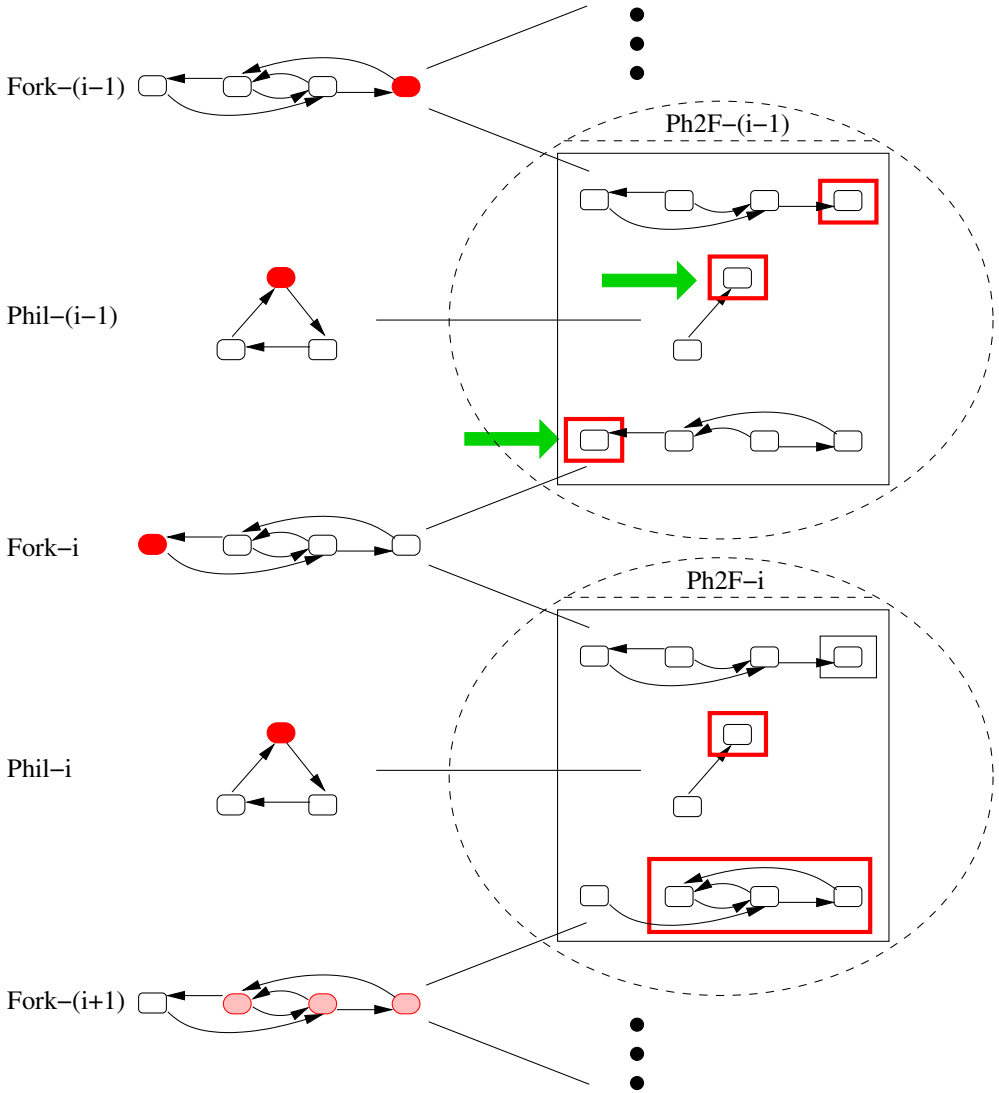
animation of as-is coordination

consistency rule

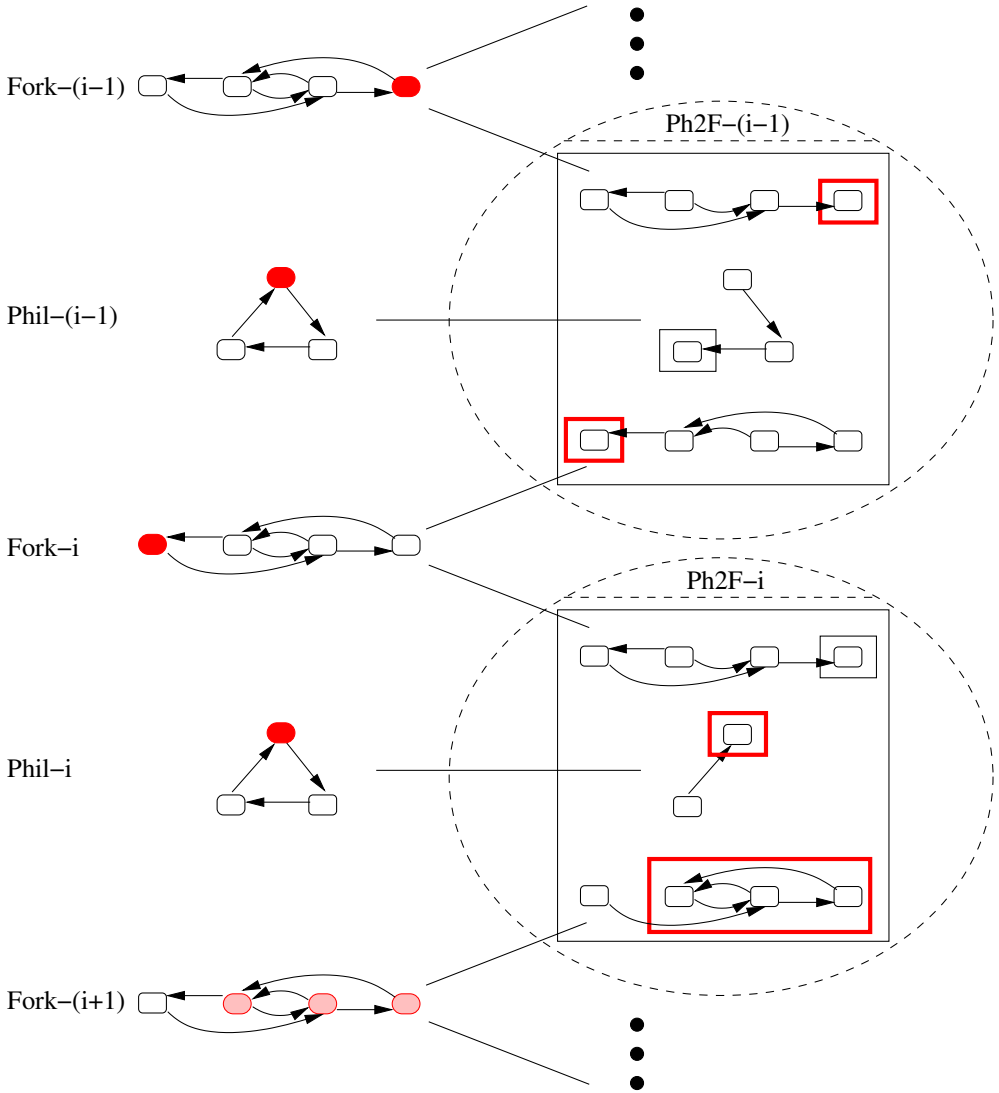
* $Fork_{i+1}(ForRH)$: $Claimed \xrightarrow{\text{got}} Claimed$,
  $Phil_i(Eater)$: $Disallowed \xrightarrow{\text{request}} Allowed$

is now applicable

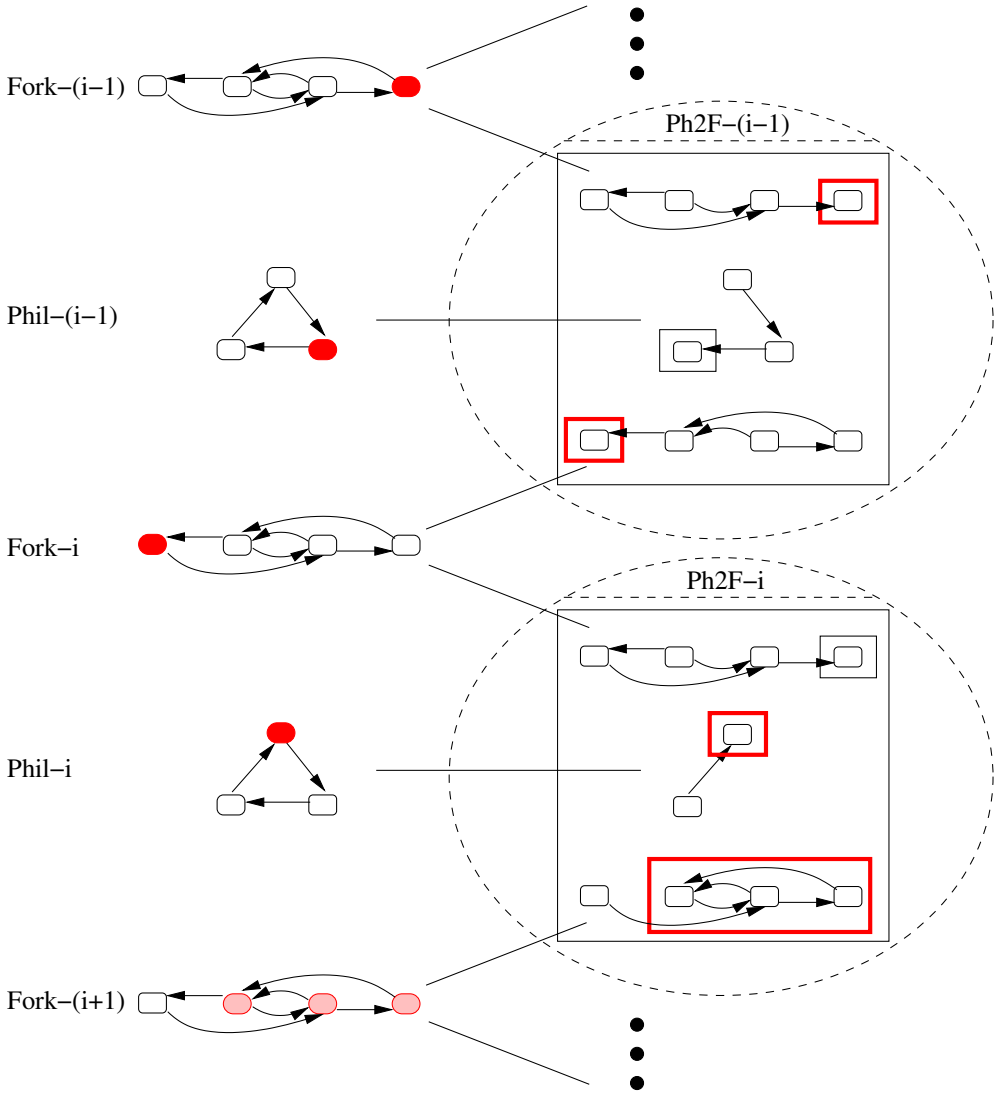
animation of as-is coordination



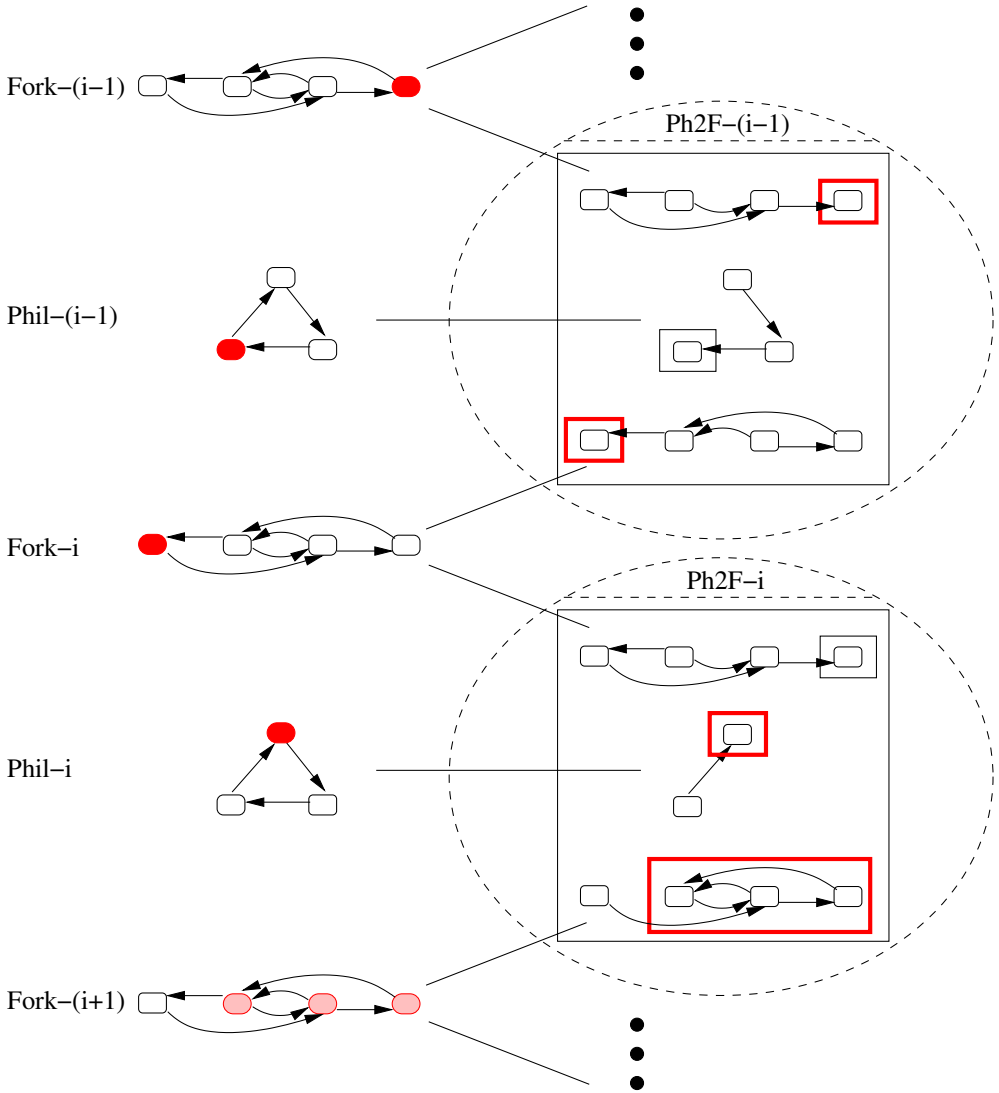
animation of as-is coordination



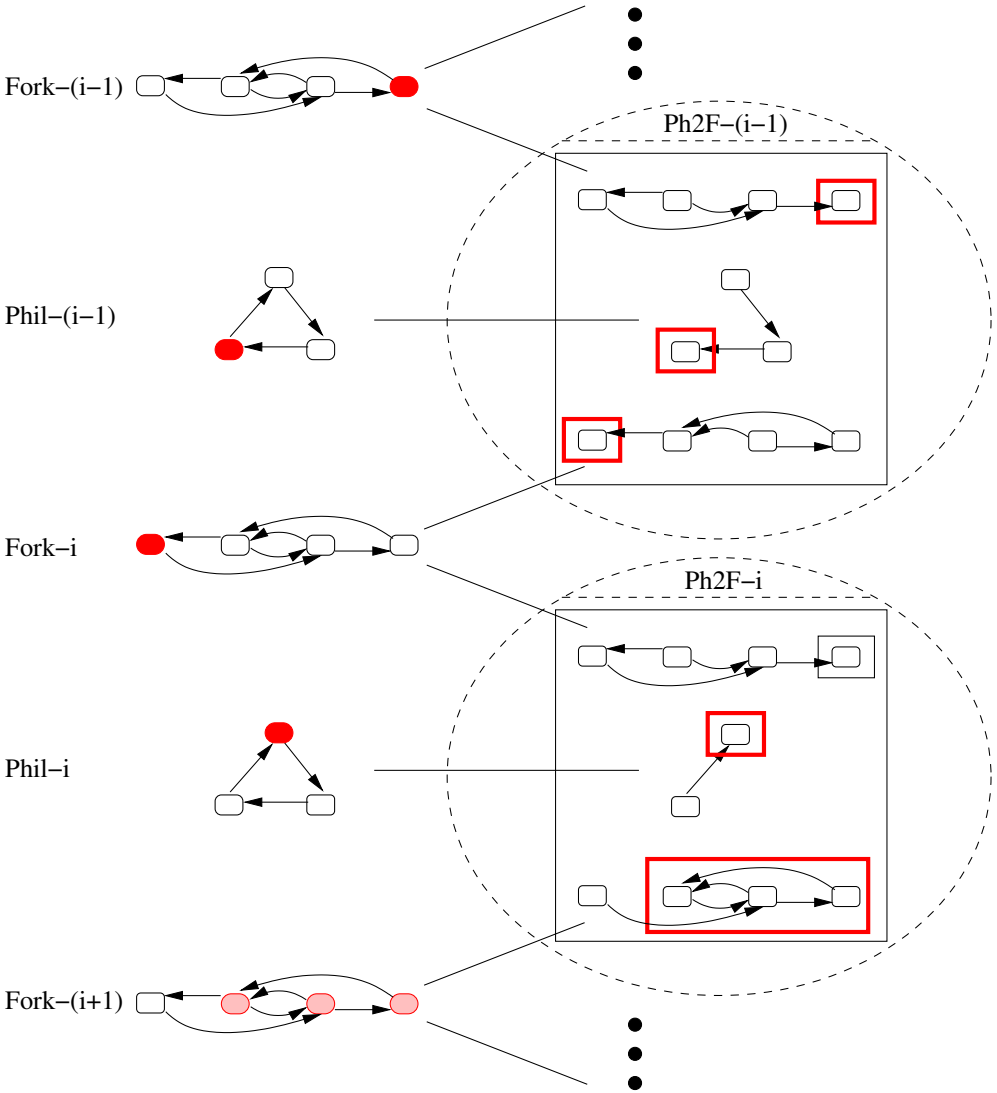
animation of as-is coordination



animation of as-is coordination



animation of as-is coordination



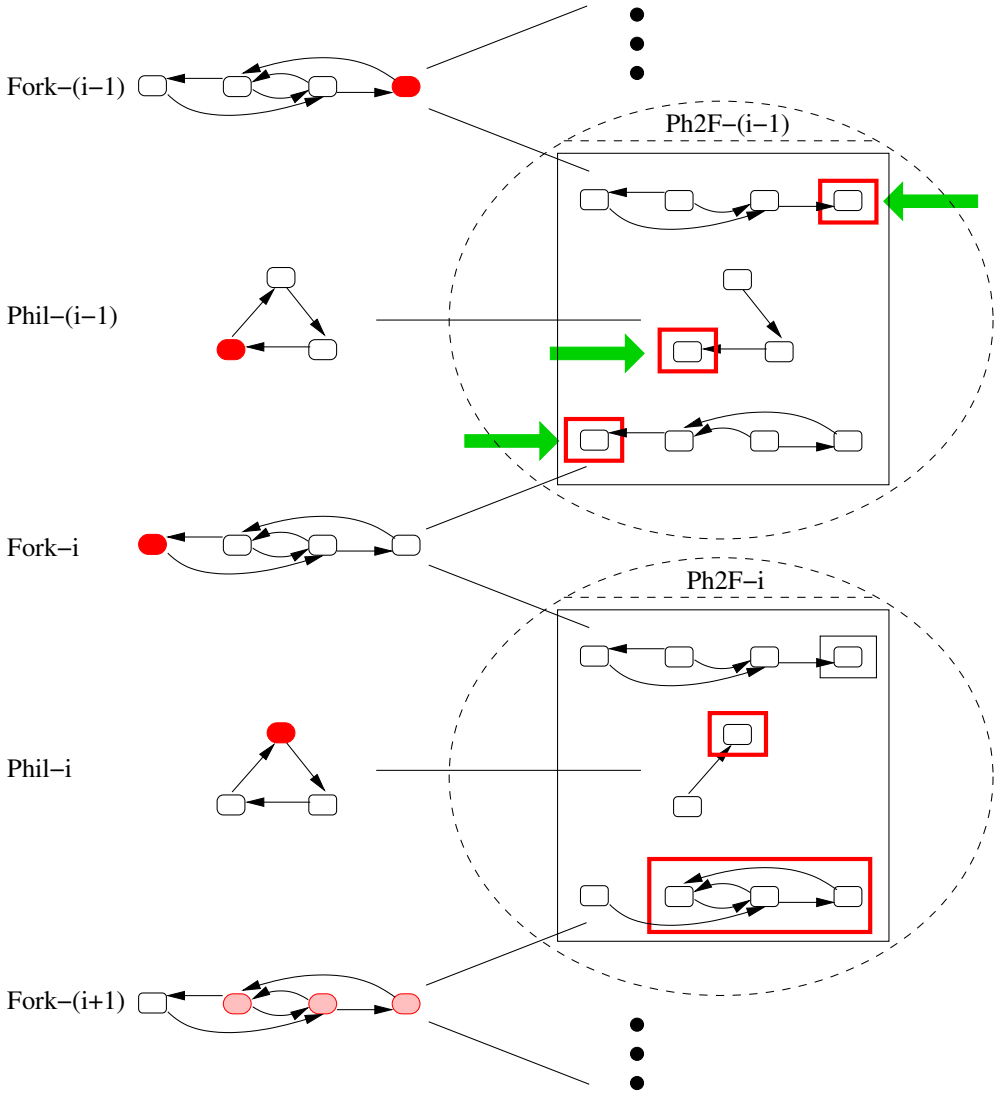
animation of as-is coordination

consistency rule

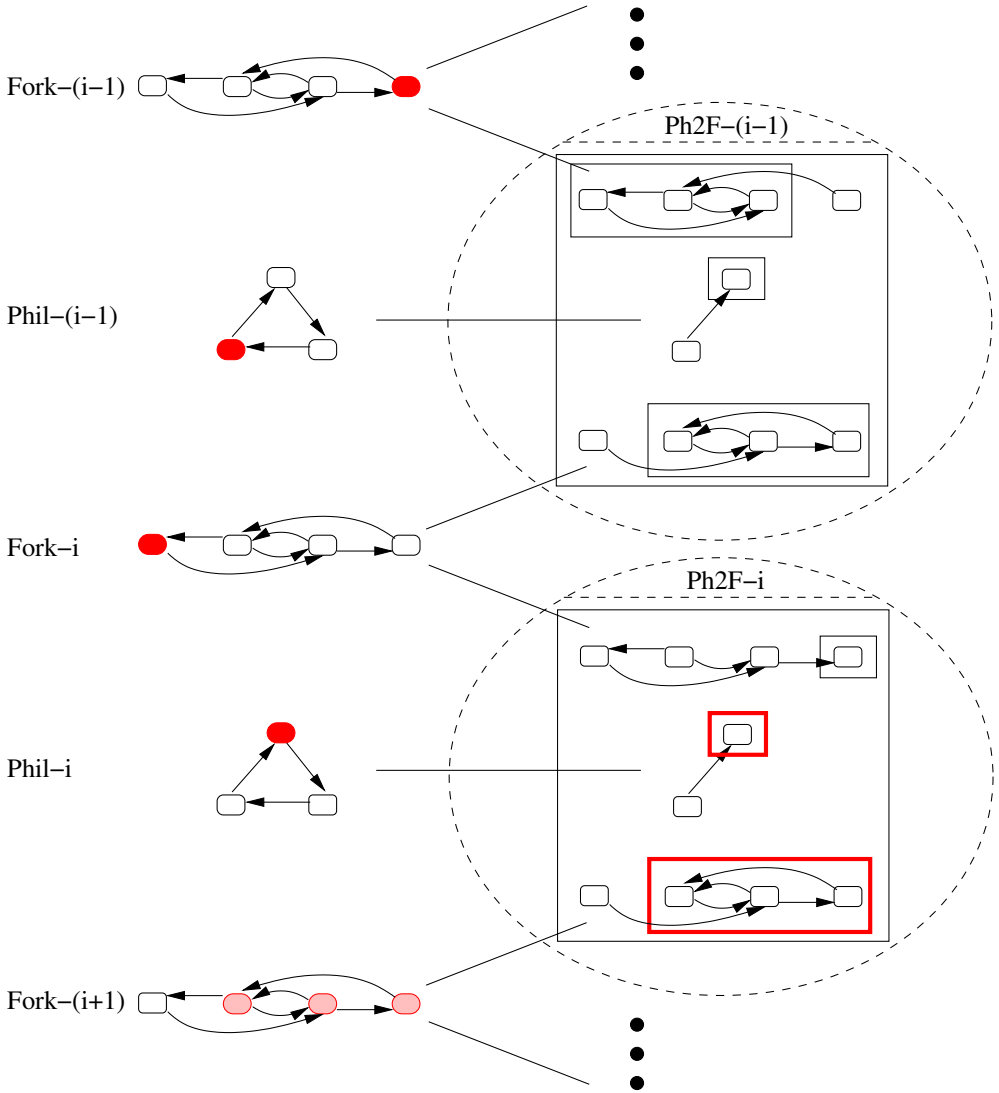
* $Phil_i(Eater)$: $Allowed \xrightarrow{\text{done}} Disallowed$,
  $Fork_i(ForLH)$: $Claimed \xrightarrow{\text{got}} Freed$,
  $Fork_{i+1}(ForRH)$: $Claimed \xrightarrow{\text{got}} Freed$

is now applicable

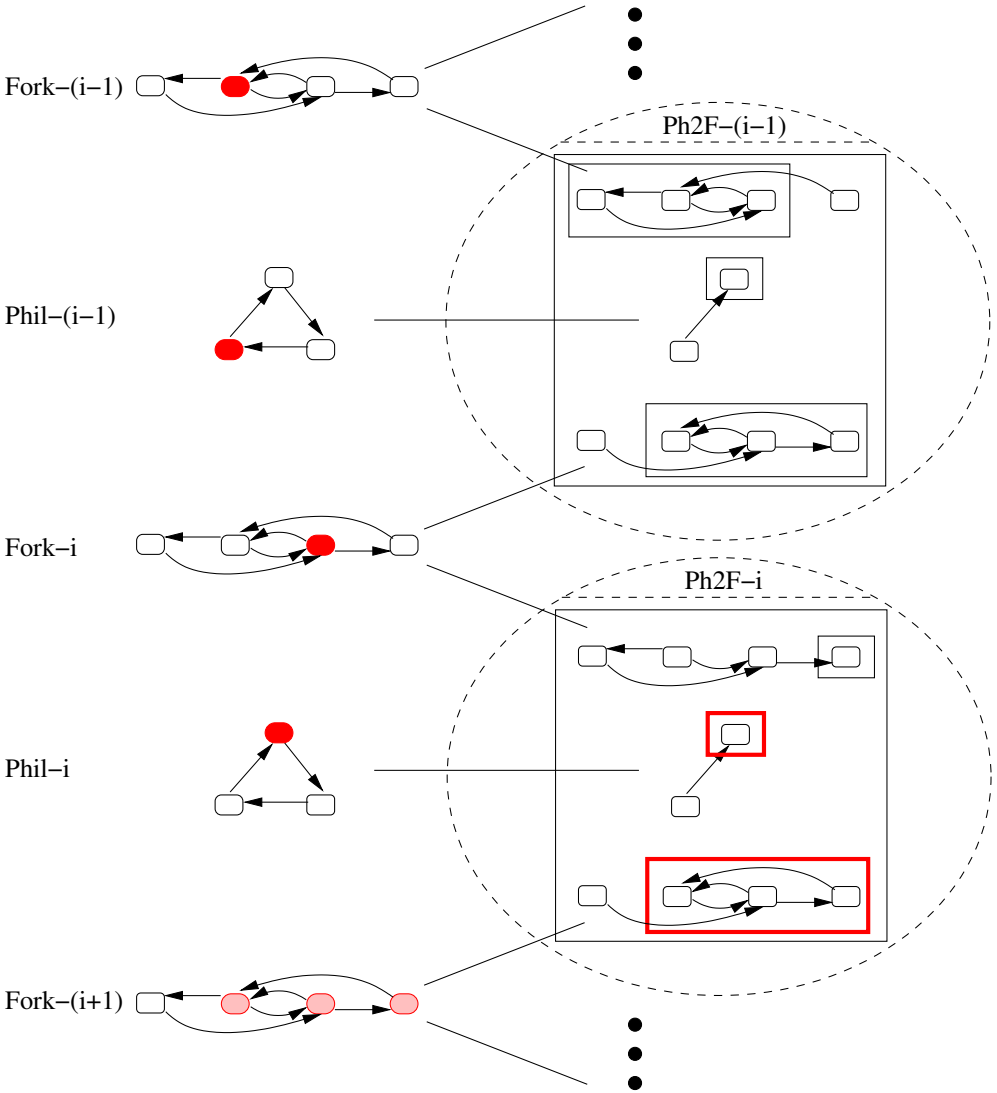
animation of as-is coordination



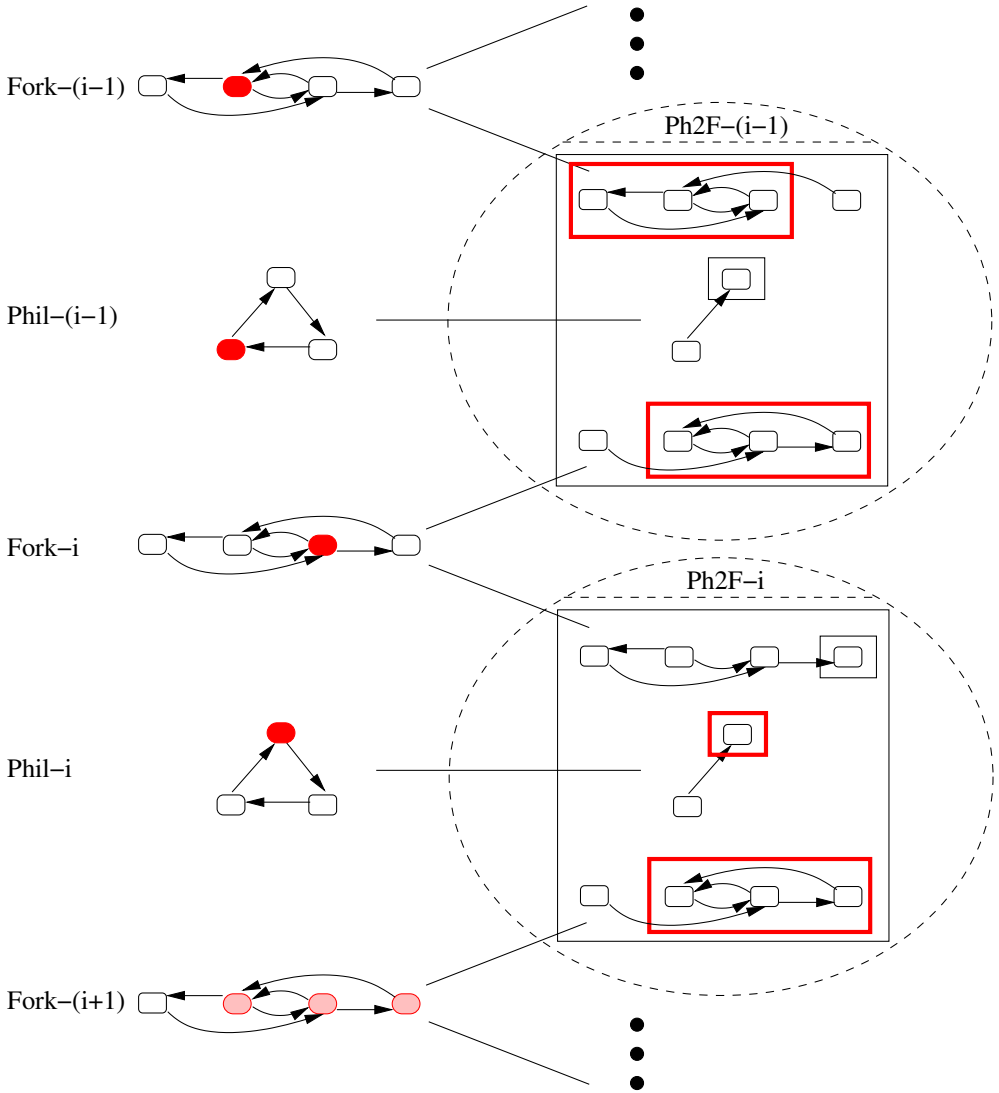
animation of as-is coordination



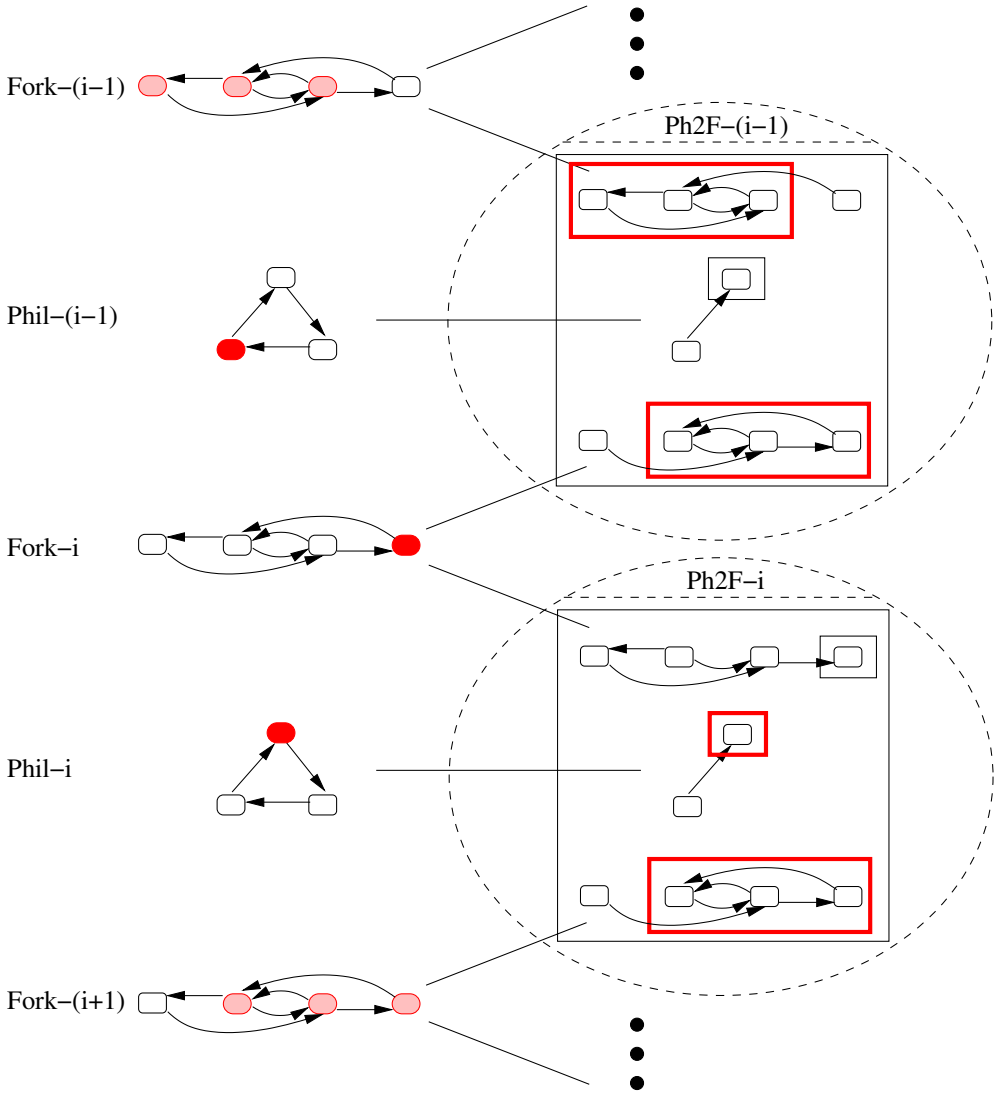
animation of as-is coordination



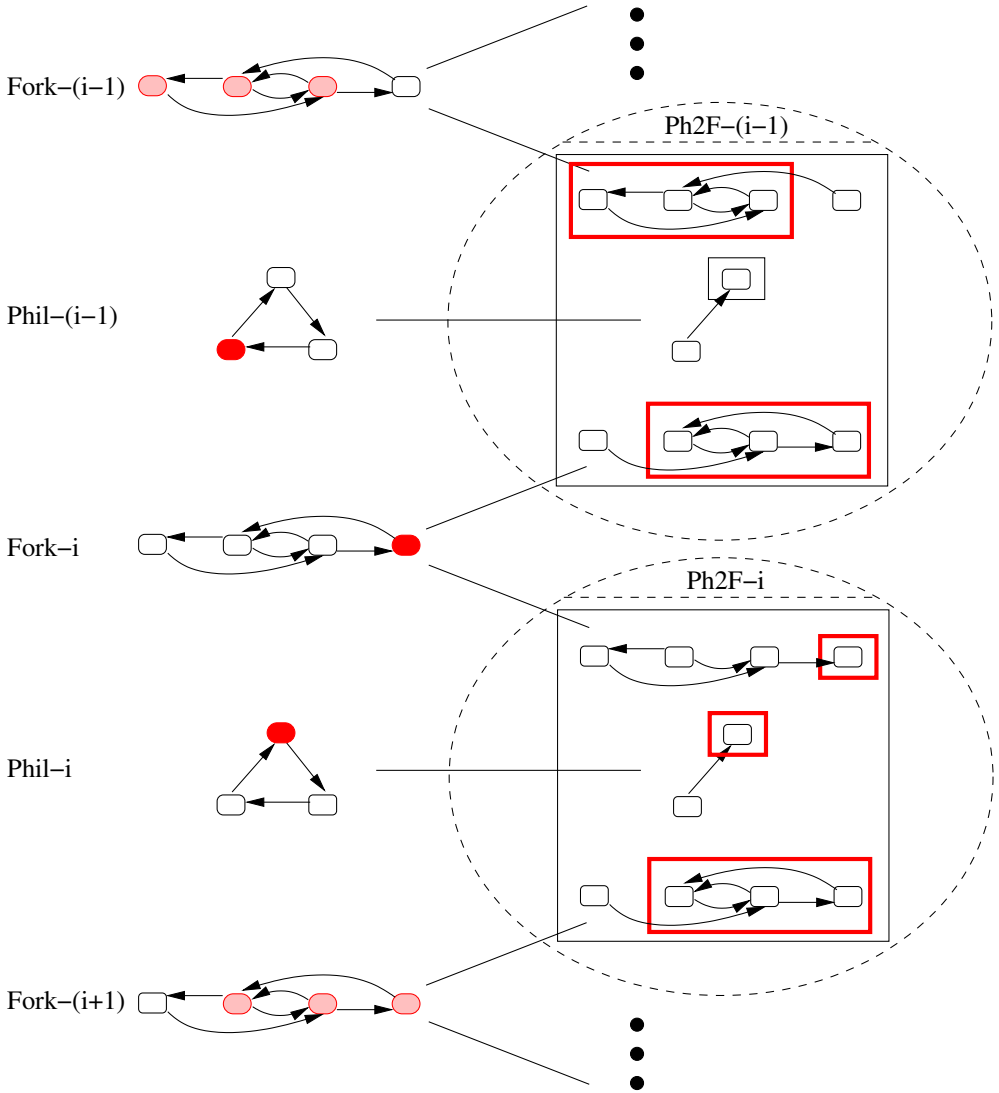
animation of as-is coordination



animation of as-is coordination



animation of as-is coordination



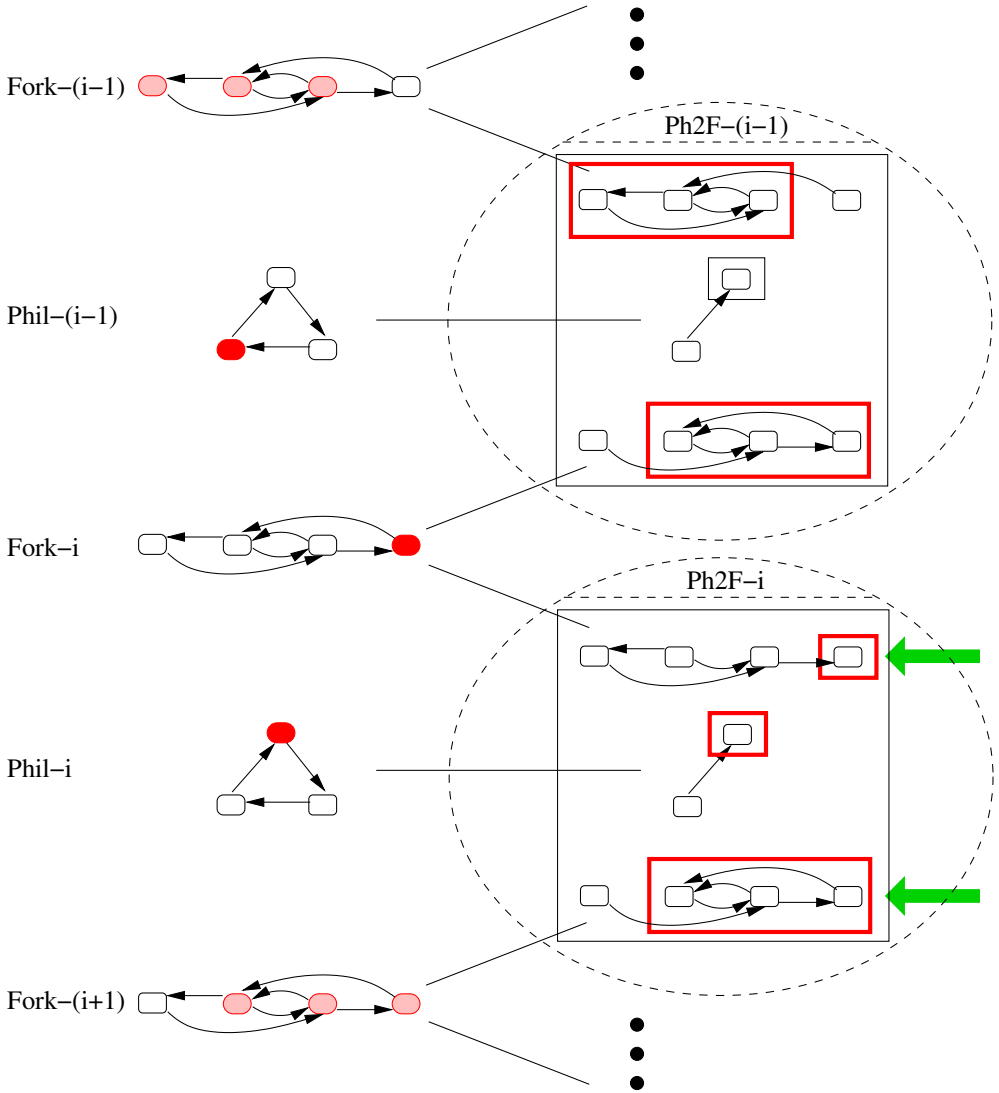
animation of as-is coordination

consistency rule

$$* \quad \mathit{Fork}_i(\mathit{ForLH}) \colon \mathit{Claimed} \xrightarrow{\text{got}} \mathit{Claimed},$$
$$\mathit{Fork}_{i+1}(\mathit{ForRH}) \colon \mathit{Freed} \xrightarrow{\text{gone}} \mathit{Claimed}$$

is now applicable

animation of as-is coordination

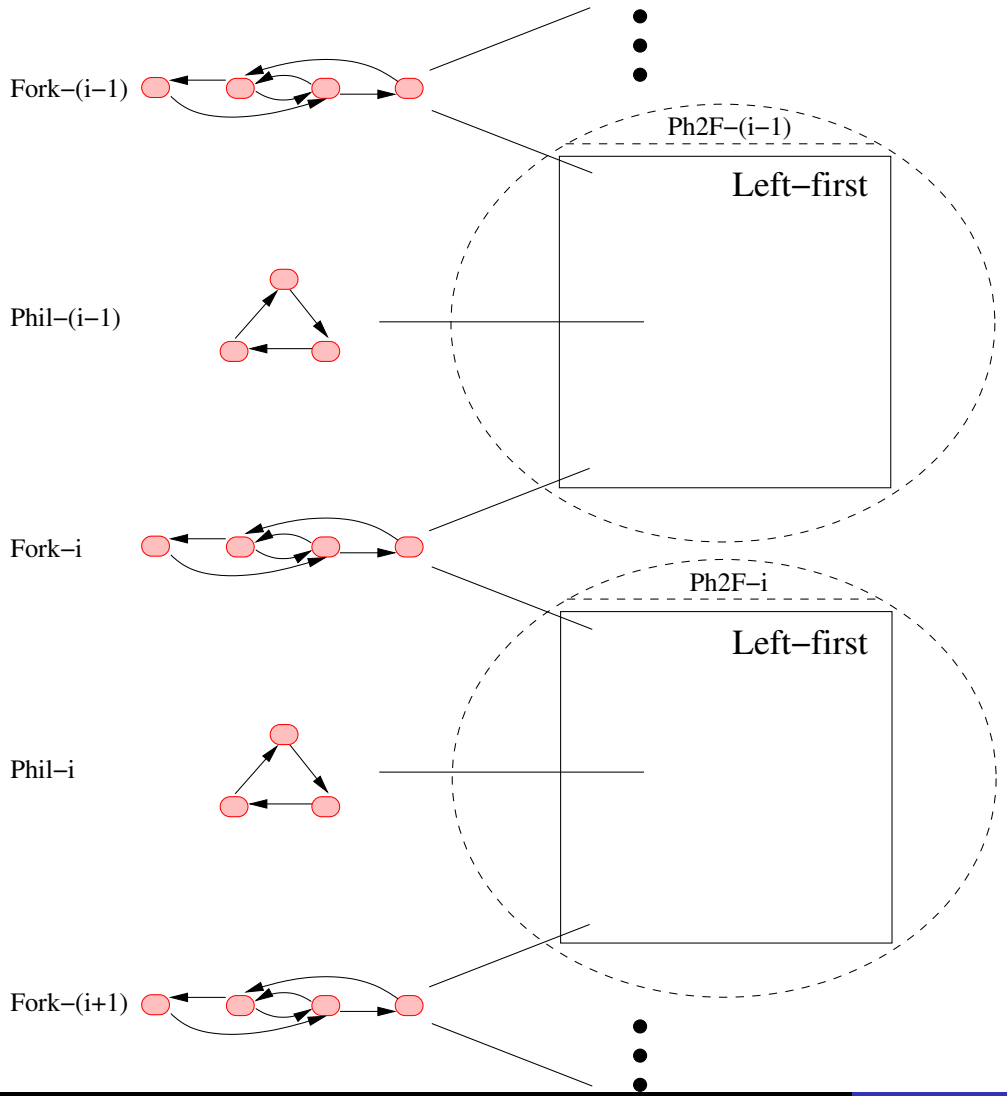


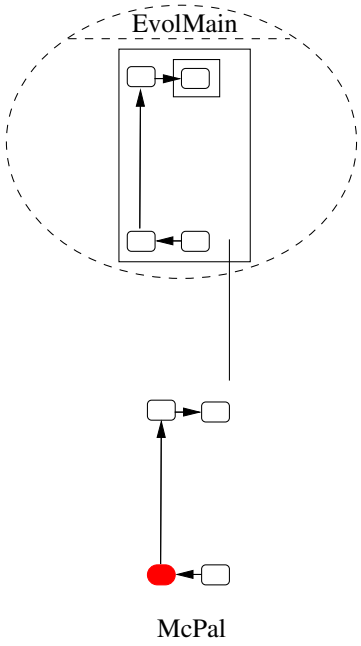
animation of migration coordination

. . . and all the time

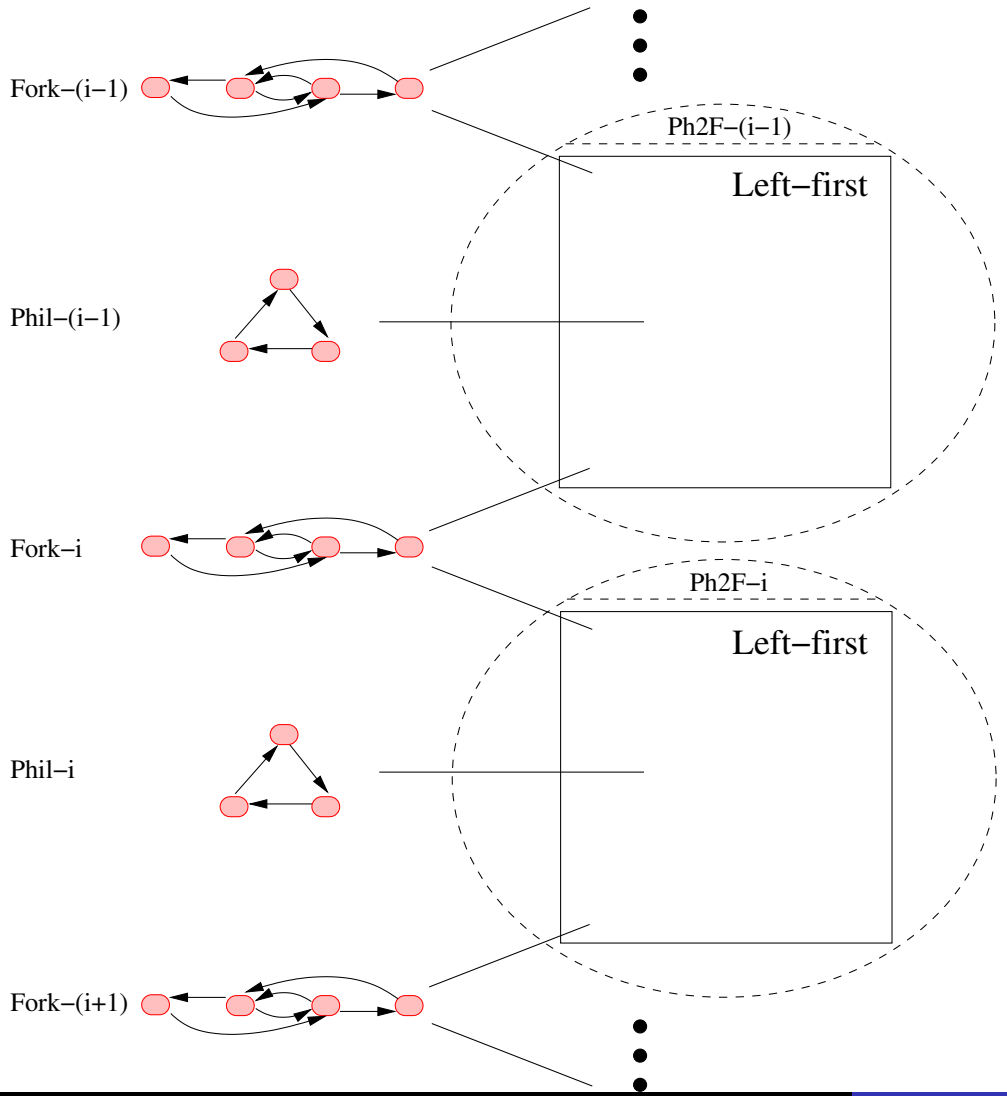
McPal is in hibernation

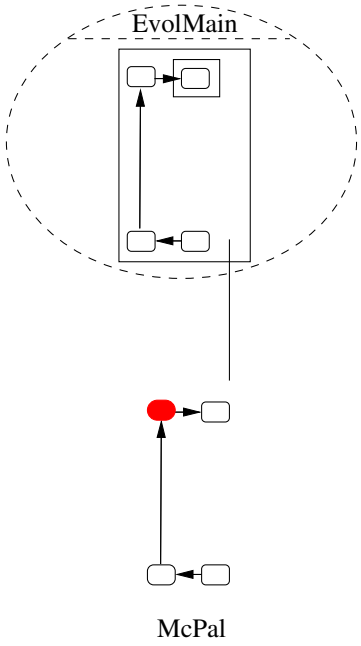
animation of migration coordination





animation of migration coordination



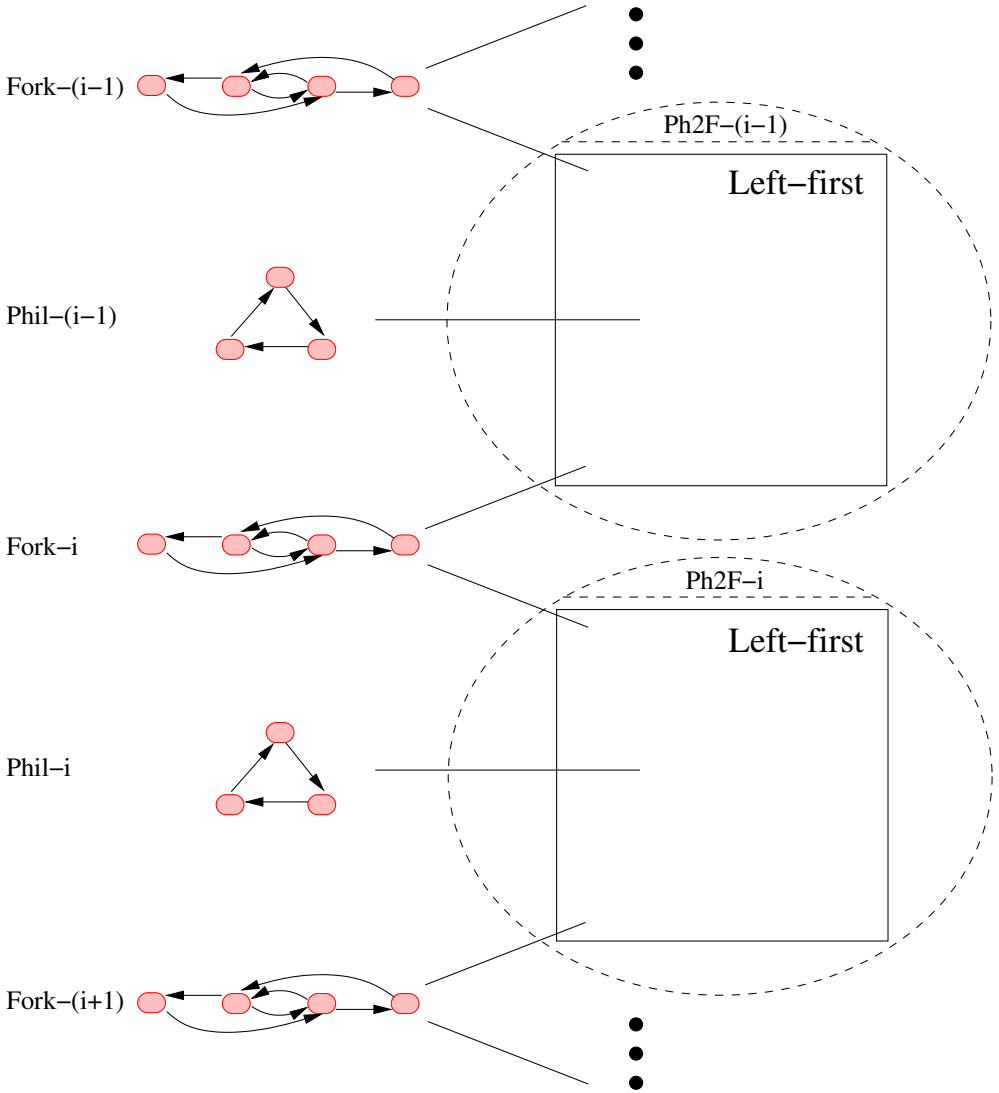


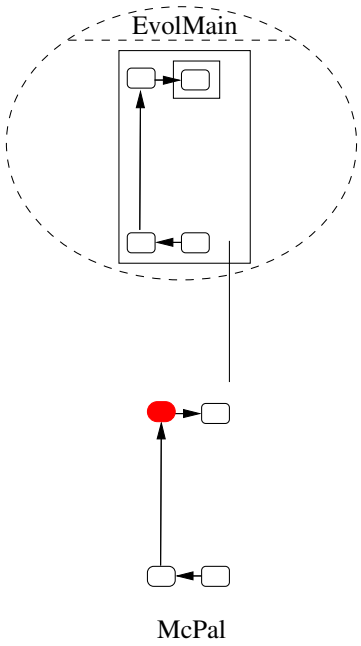
animation of migration coordination

though hibernating,

McPal prepares an adaptation

animation of migration coordination





animation of migration coordination

and *McPal* decides to achieve for to-be situation again

5 $Phil2Forks_i$ collaborations

but not LeftFirst for each

animation of migration coordination

# of these 5 collaborations

- LeftFirst for 3
- RightFirst for 2

animation of migration coordination

of these 5 collaborations

- LeftFirst for 3 or for 2
- RightFirst for 2 or for 3

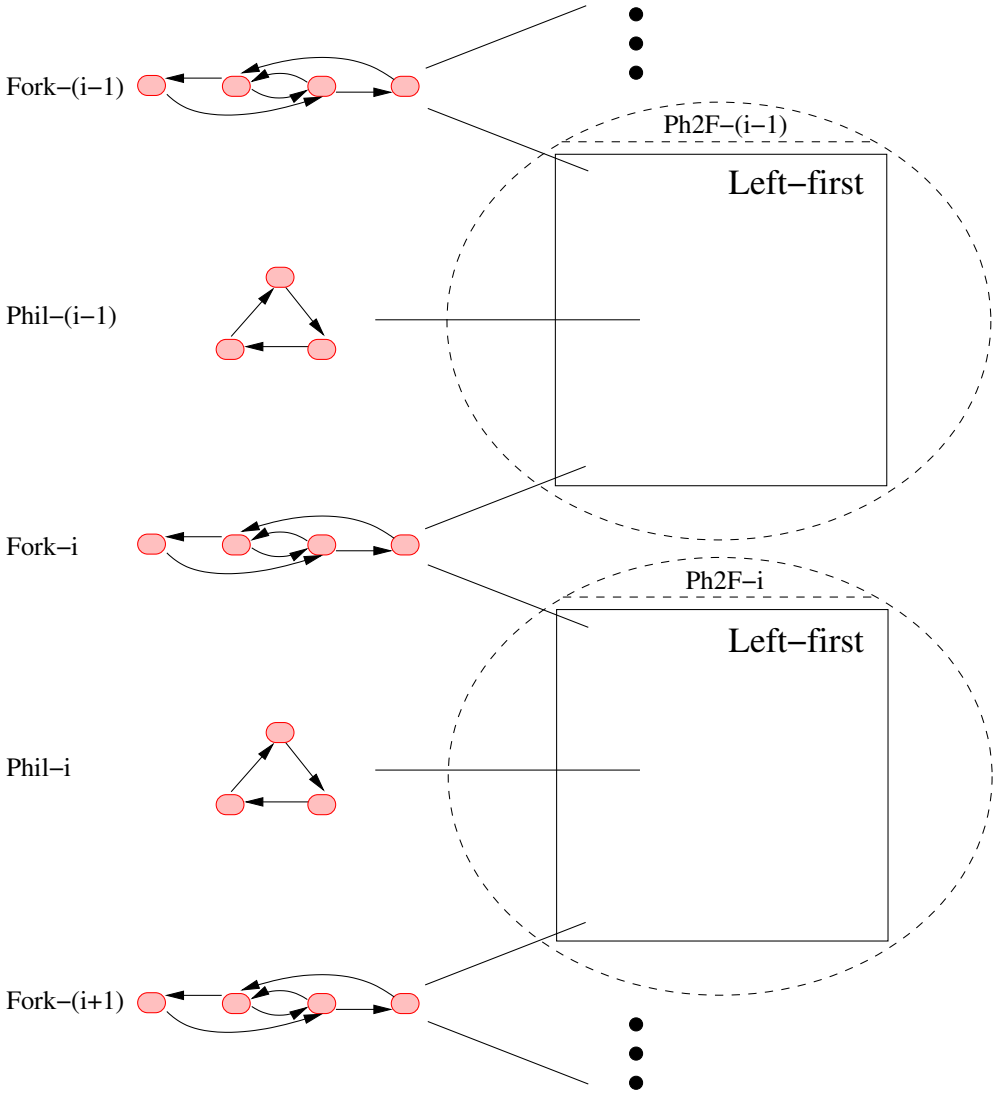
animation of migration coordination

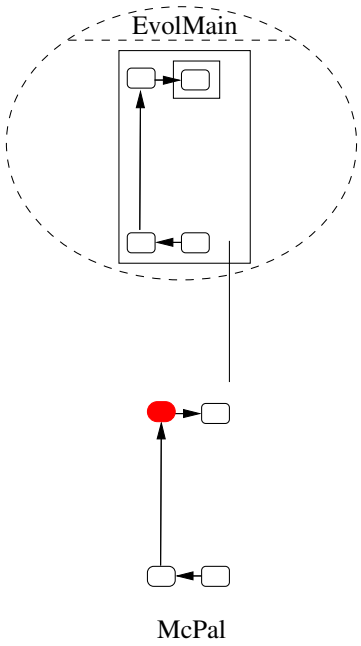
of these 5 collaborations

- LeftFirst for 3 or for 2
- RightFirst for 2 or for 3

- the 2 or 2 aren't neighbours

animation of migration coordination





animation of migration coordination

once *McPal* has clear
what to do and how,
McPal extends the model

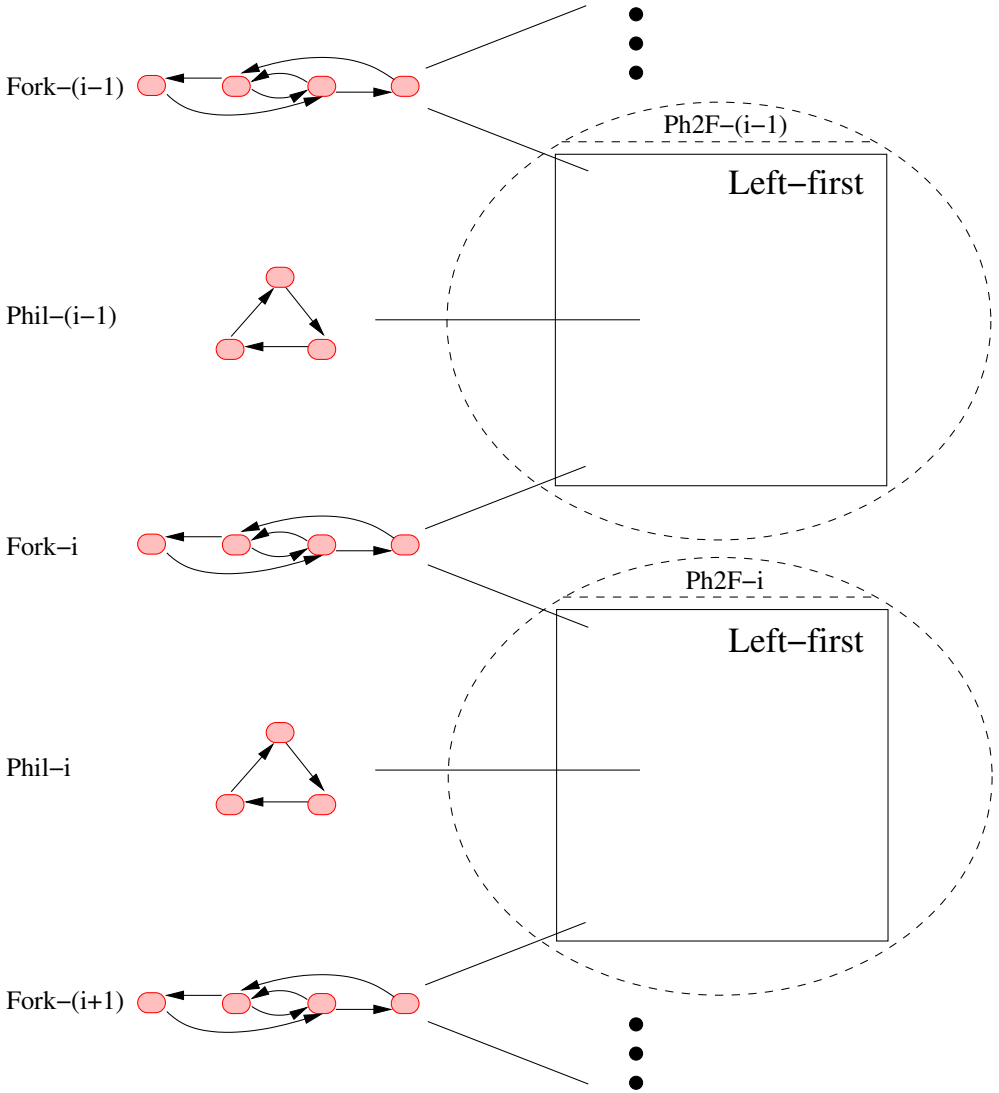
while keeping all constraints

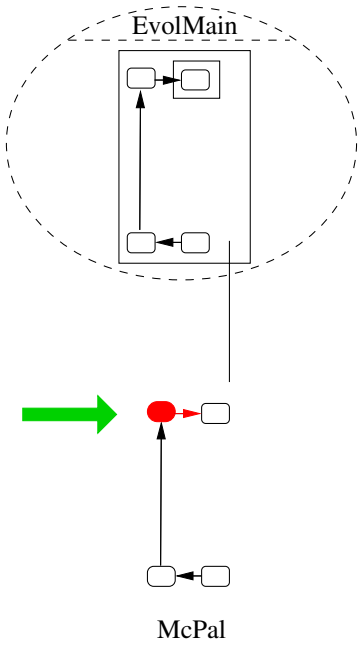
animation of migration coordination

via consistency rule

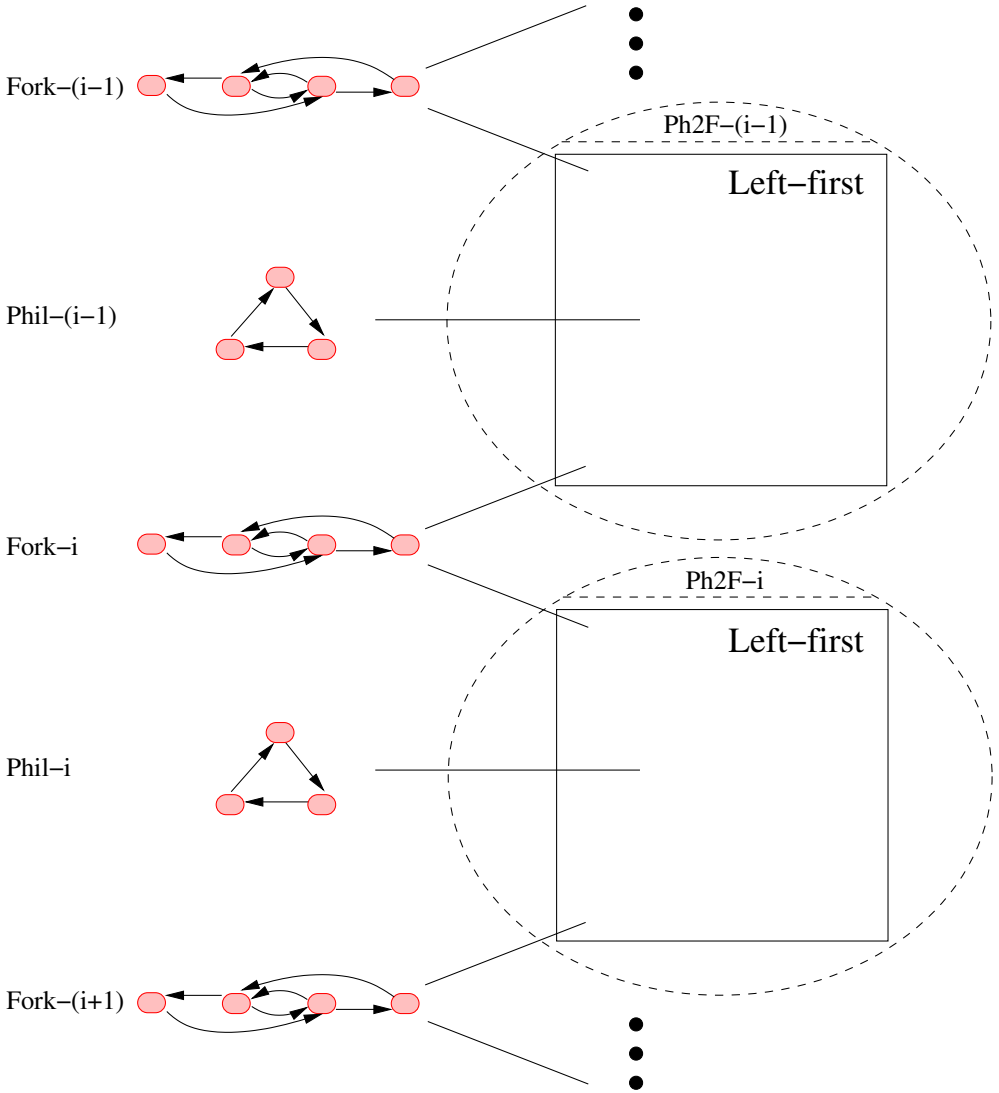
$$McPal : JITting \xrightarrow{\text{giveOut}} StartMigr \; *$$
$$McPal : [Crs := Crs + Crs_{migr} + Crs_{toBe}]$$

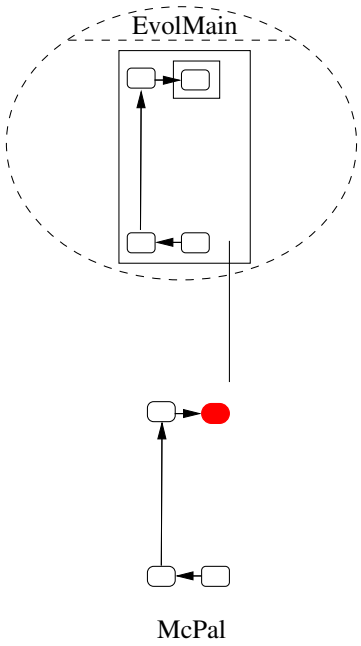
animation of migration coordination





animation of migration coordination



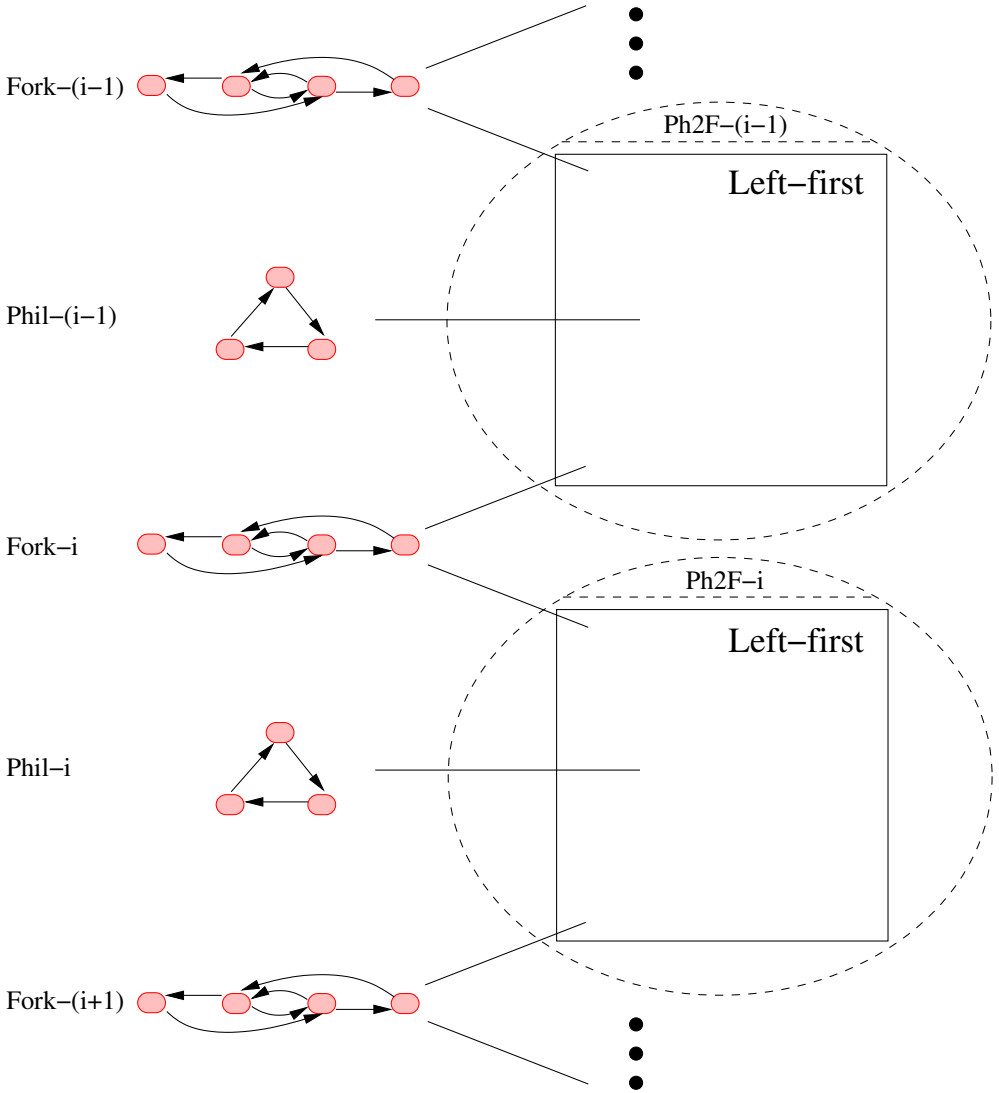


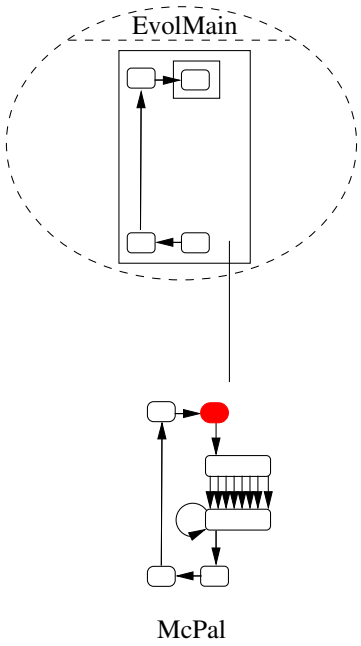
animation of migration coordination

so, more description is in place

but dynamics remains as-is

animation of migration coordination



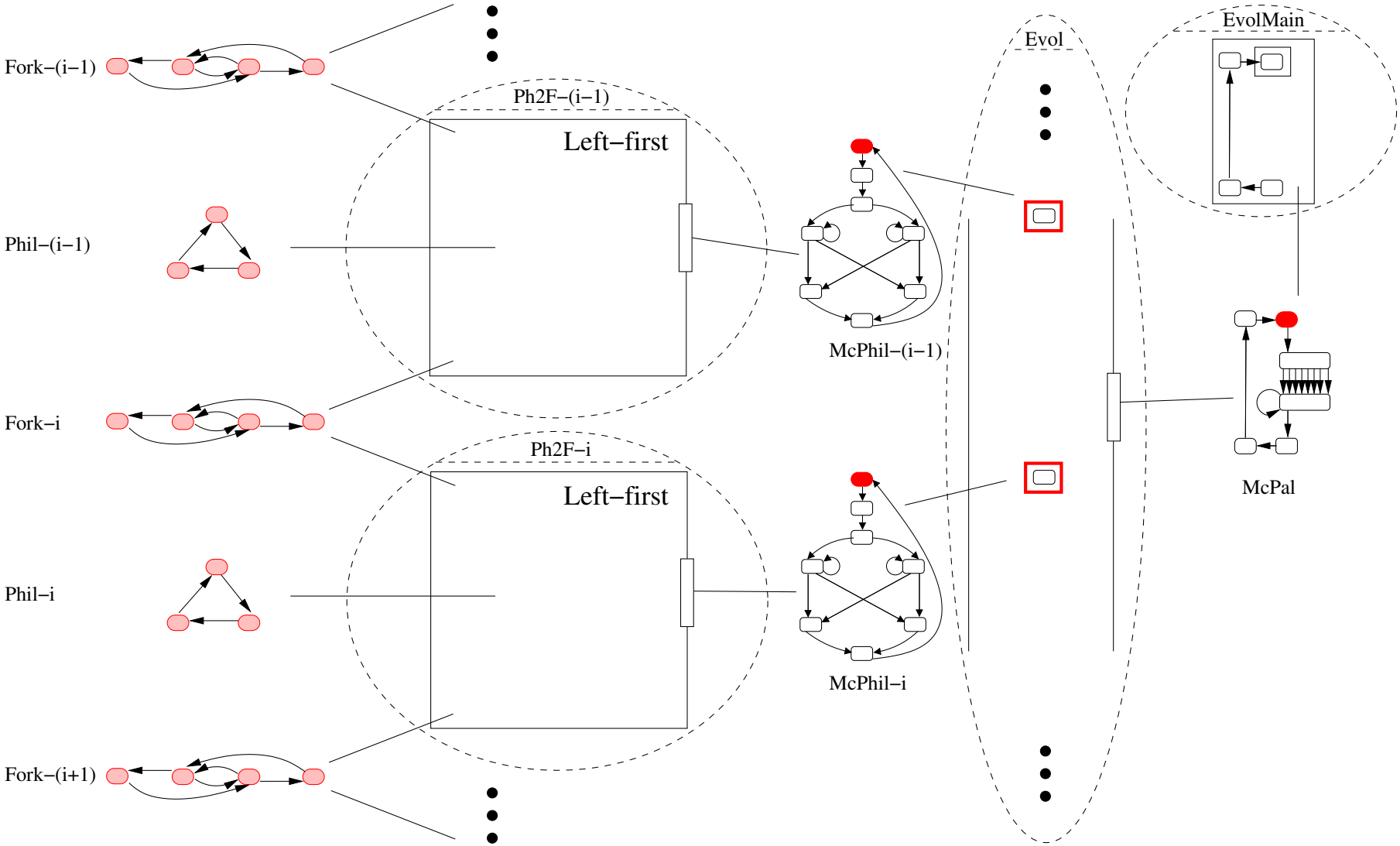


animation of migration coordination

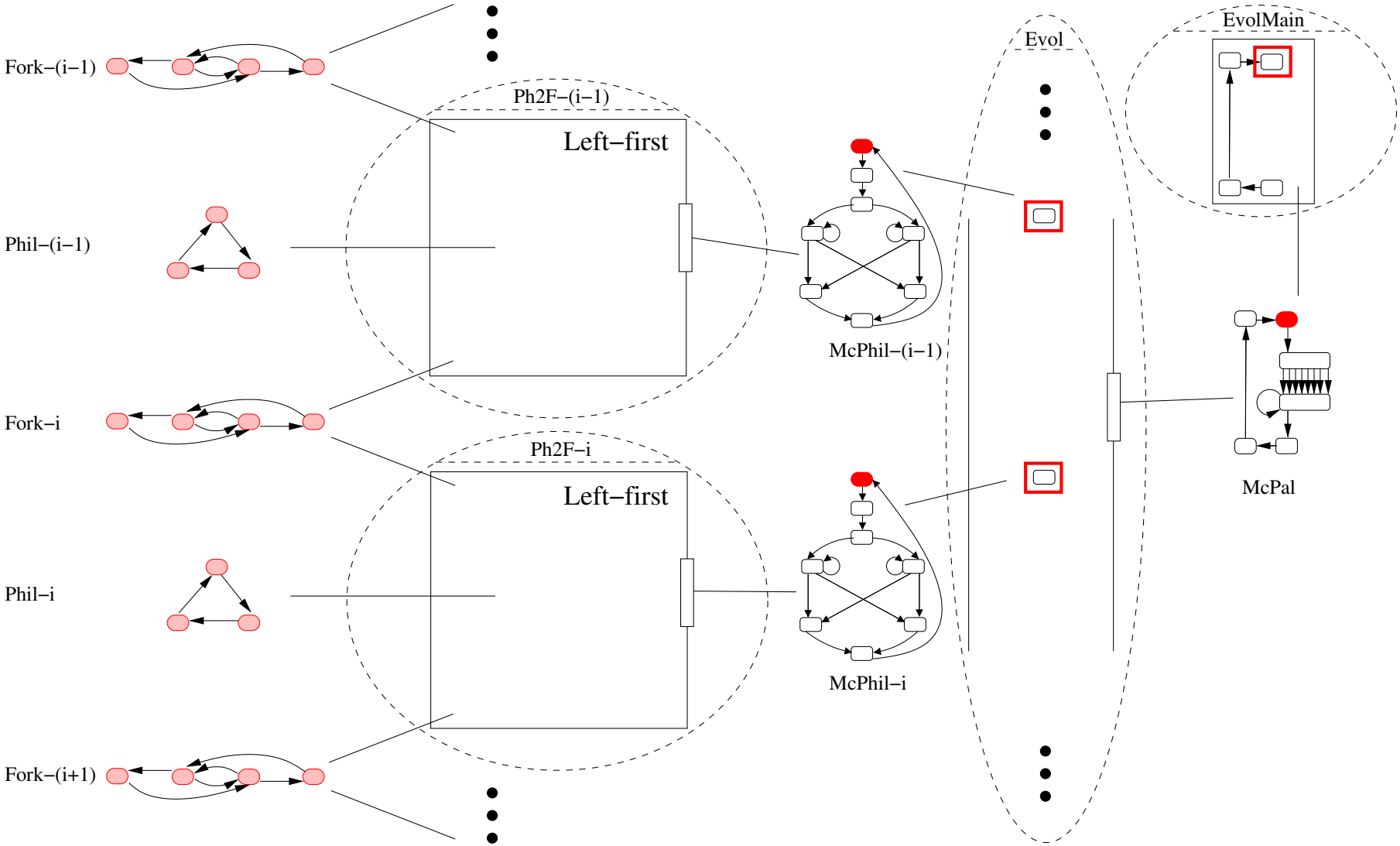
and more description is in place

while dynamics remains as-is

animation of migration coordination



animation of migration coordination



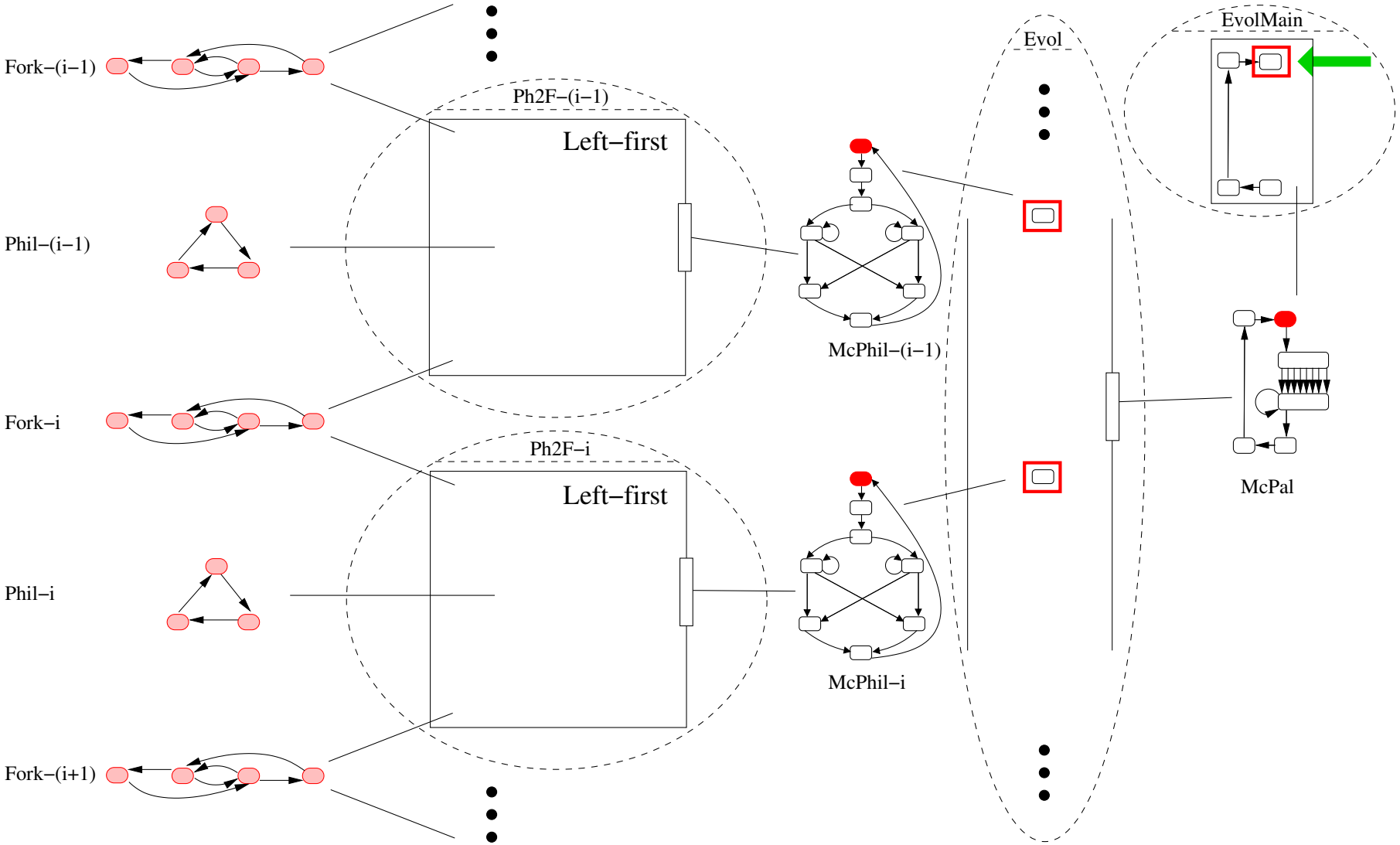
animation of migration coordination

via choreography rule

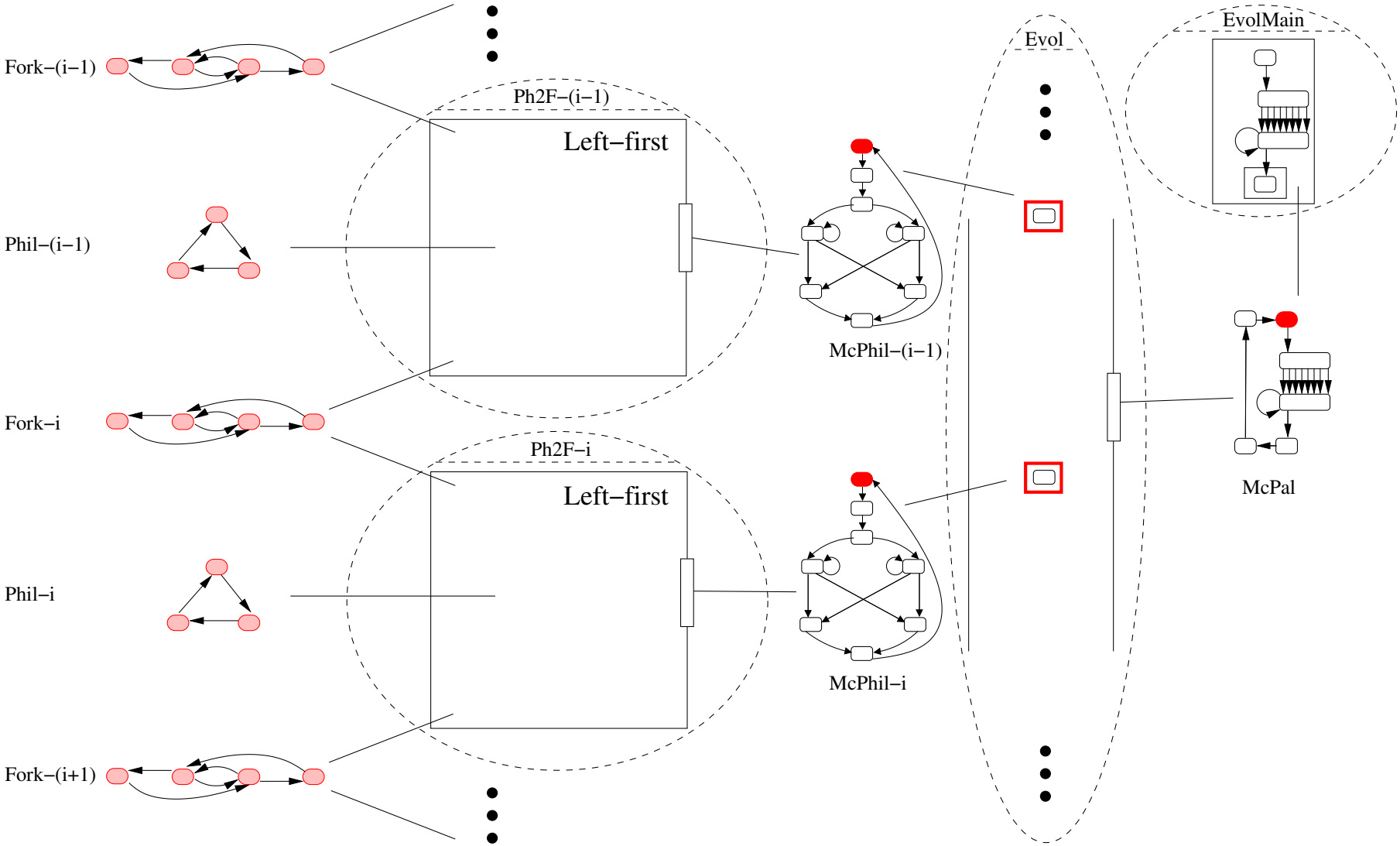
$*$ $McPal(Evol) : Hibernating \xrightarrow{\text{prepared}} Migrating$

first *McPal*'s own migration begins

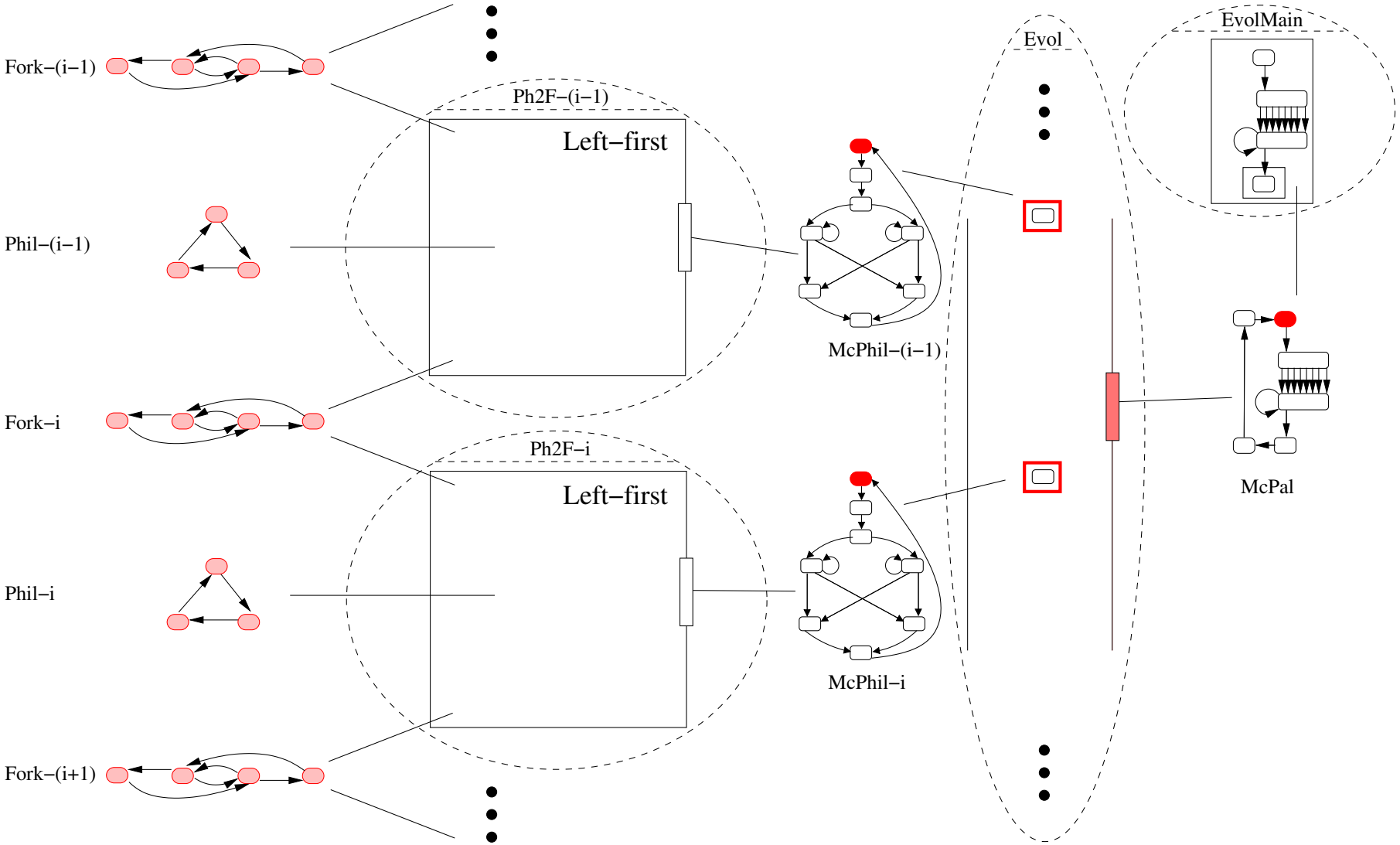
animation of migration coordination



animation of migration coordination



animation of migration coordination



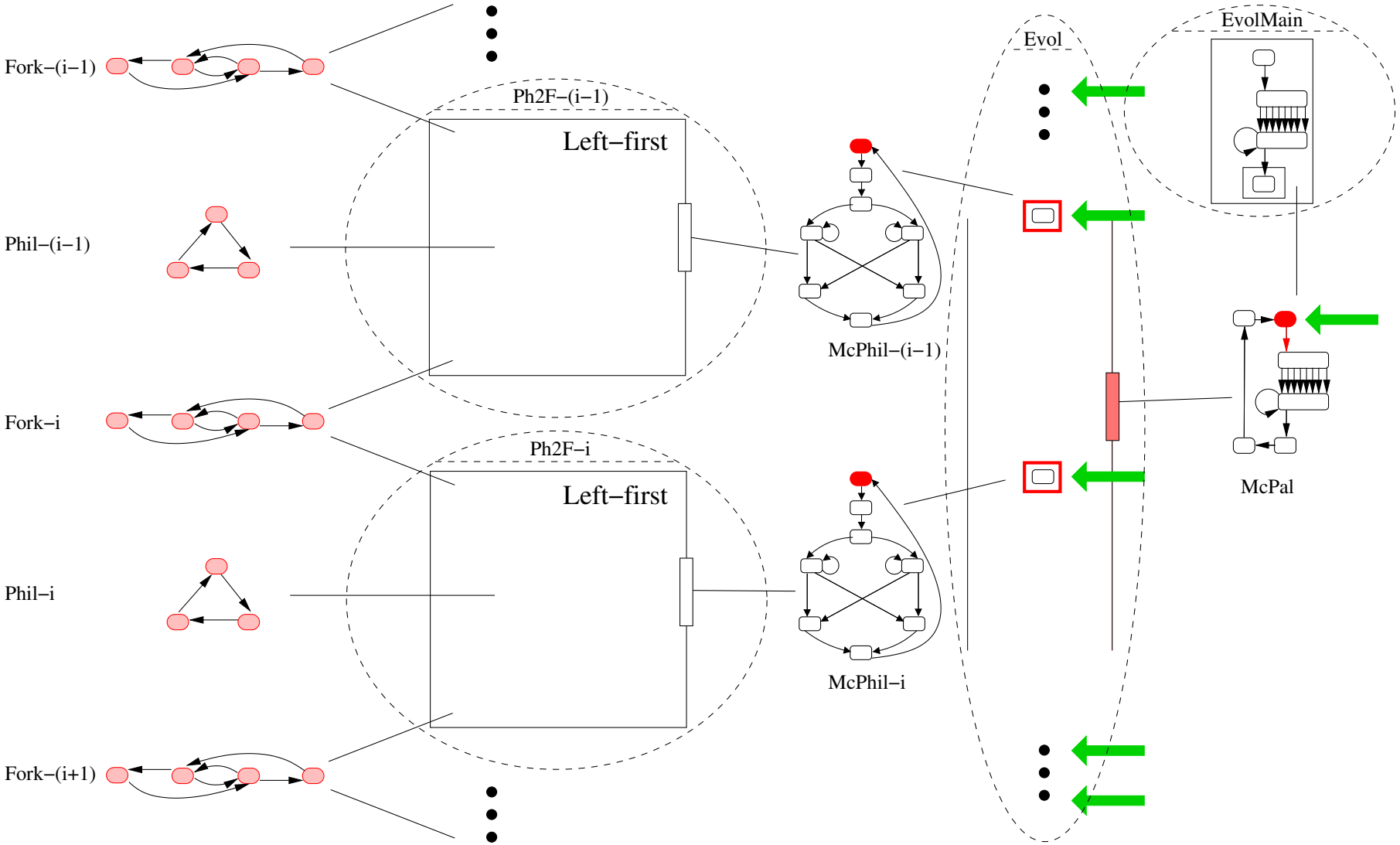
animation of migration coordination

orchestration rule

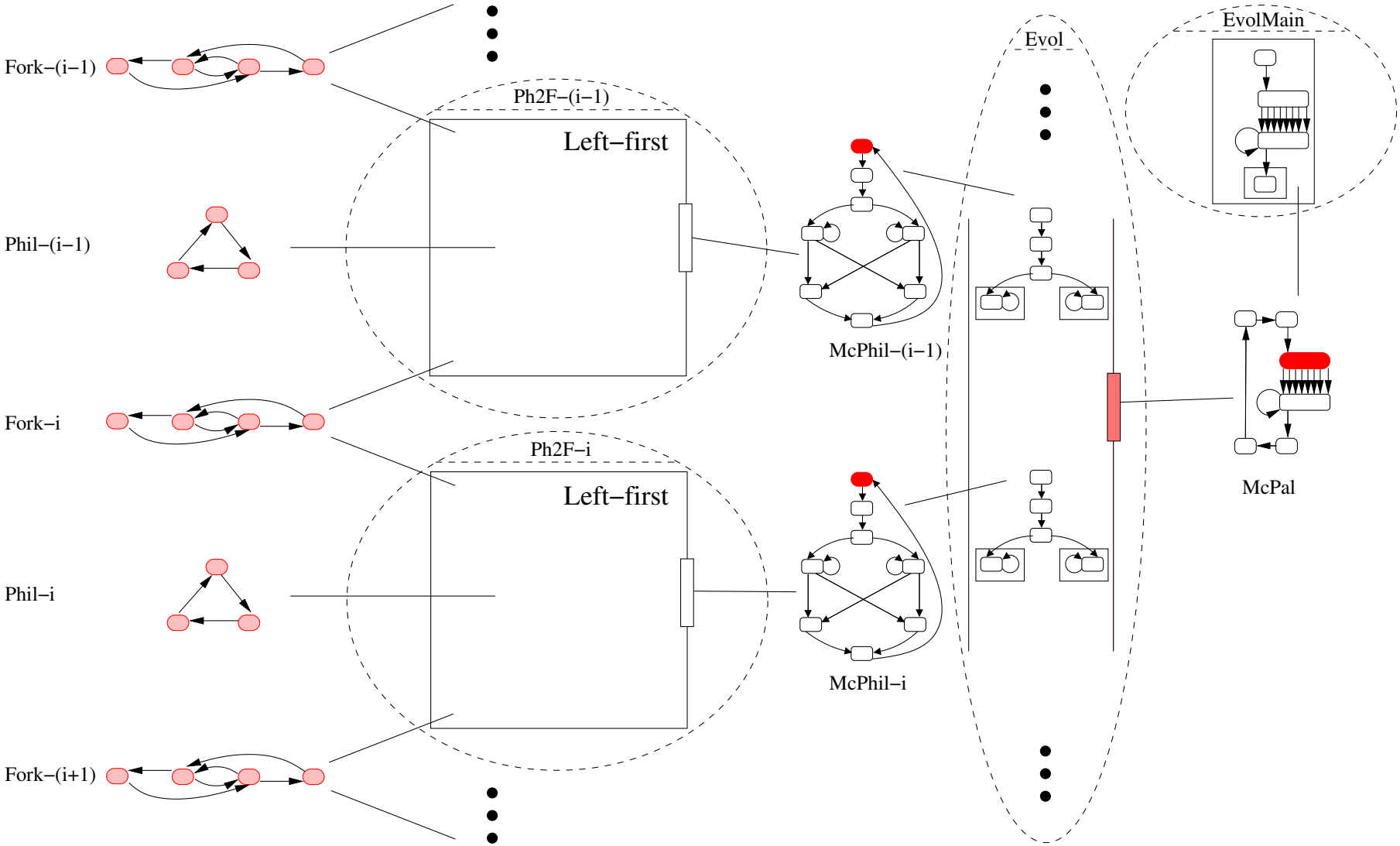
$$McPal : StartMigr \xrightarrow{\text{create}} Delegated \; *$$
$$McPhil_1(Evol) : Passive \xrightarrow{\text{triv}} Active,$$
$$McPhil_2(Evol) : Passive \xrightarrow{\text{triv}} Active, \ldots$$
$$\ldots, McPhil_5(Evol) : Passive \xrightarrow{\text{triv}} Active,$$
$$McPal : [Crs := Crs_{noHb} + Crs_{hibr}]$$
$$McPhil_1 : [Crs_1 := Crs_{1,asls} + Crs_{1,orch}], \ldots$$
$$\ldots, McPhil_5 : [Crs_5 := Crs_{5,asls} + Crs_{5,orch}]$$

is now applicable

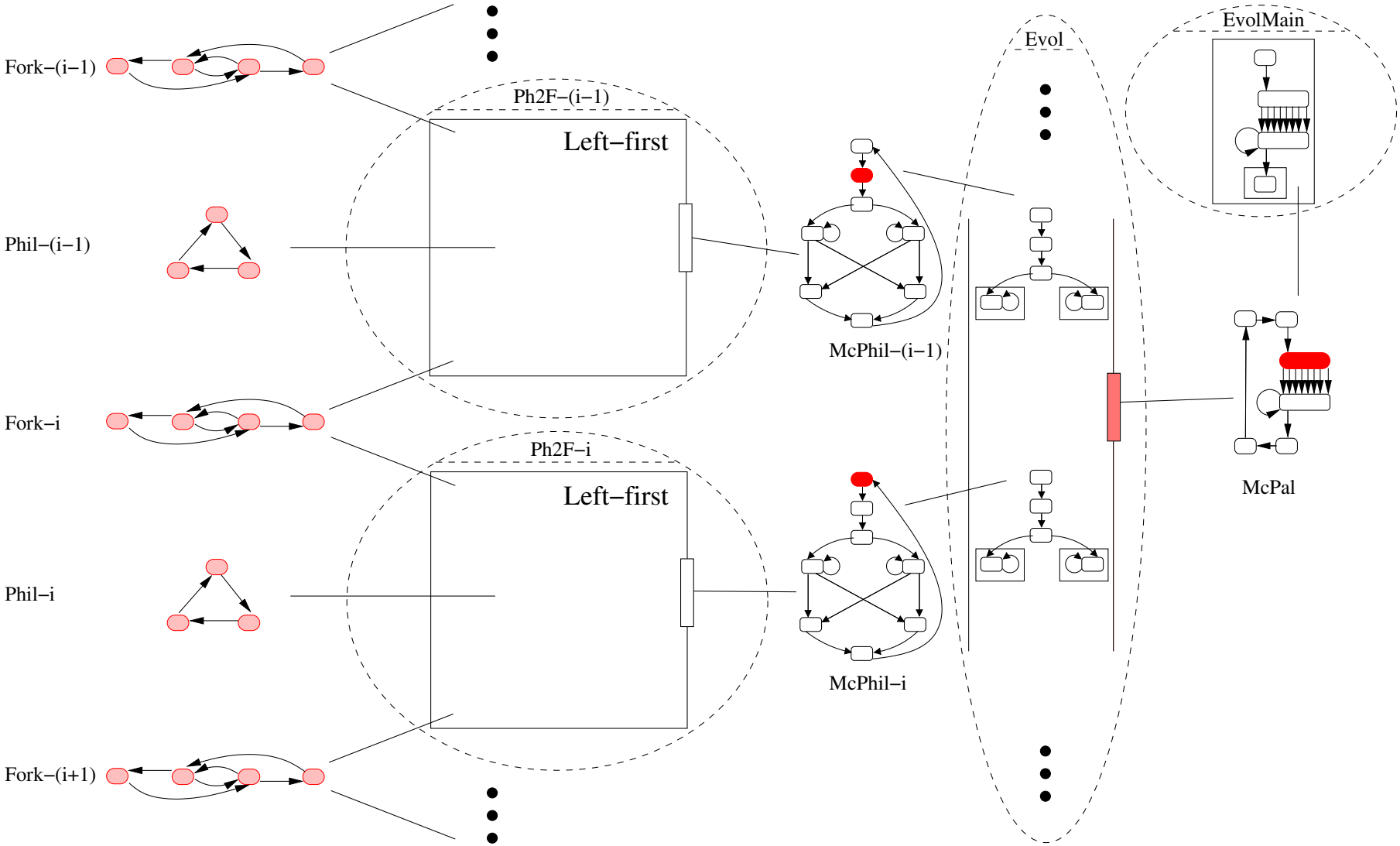
animation of migration coordination



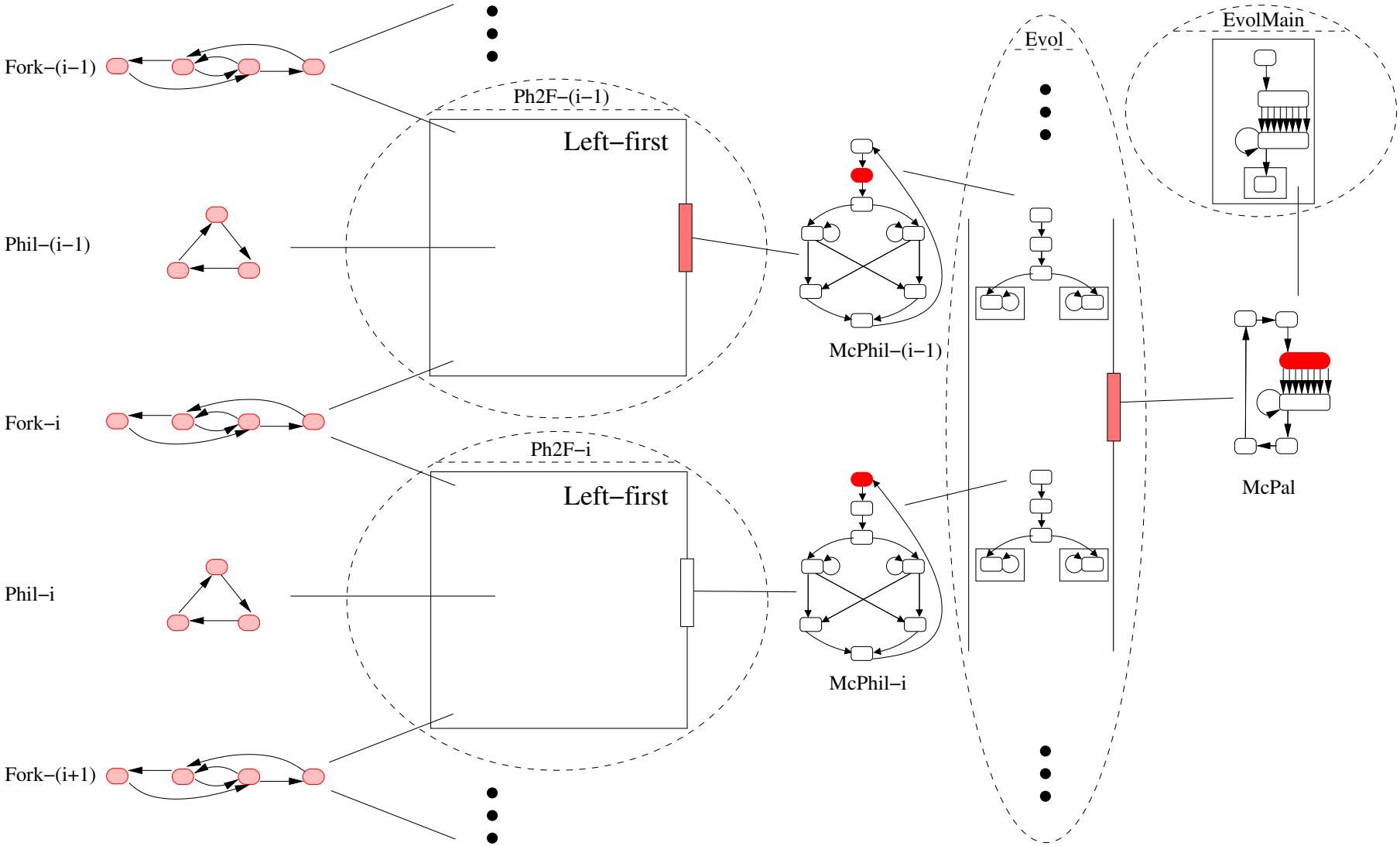
animation of migration coordination



animation of migration coordination



animation of migration coordination



animation of migration coordination

orchestration rule

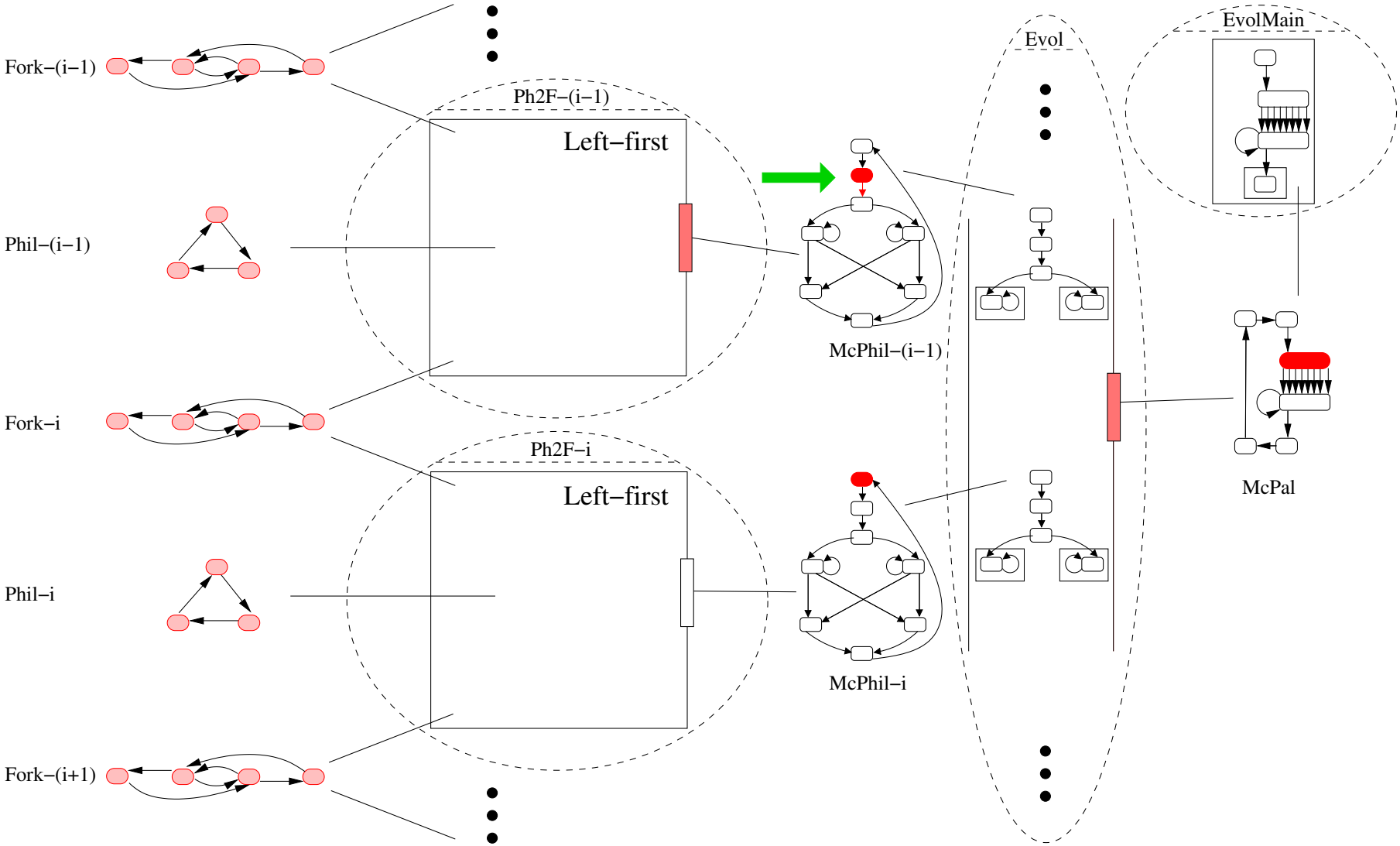
$$McPhil_{i-1} : Awake \xrightarrow{\text{takeOver}} JoiningIn * McPhil_{i-1} : [Crs_{i-1} := Crs_{i-1} - Crs_{i-1,asls}]$$

is now applicable,

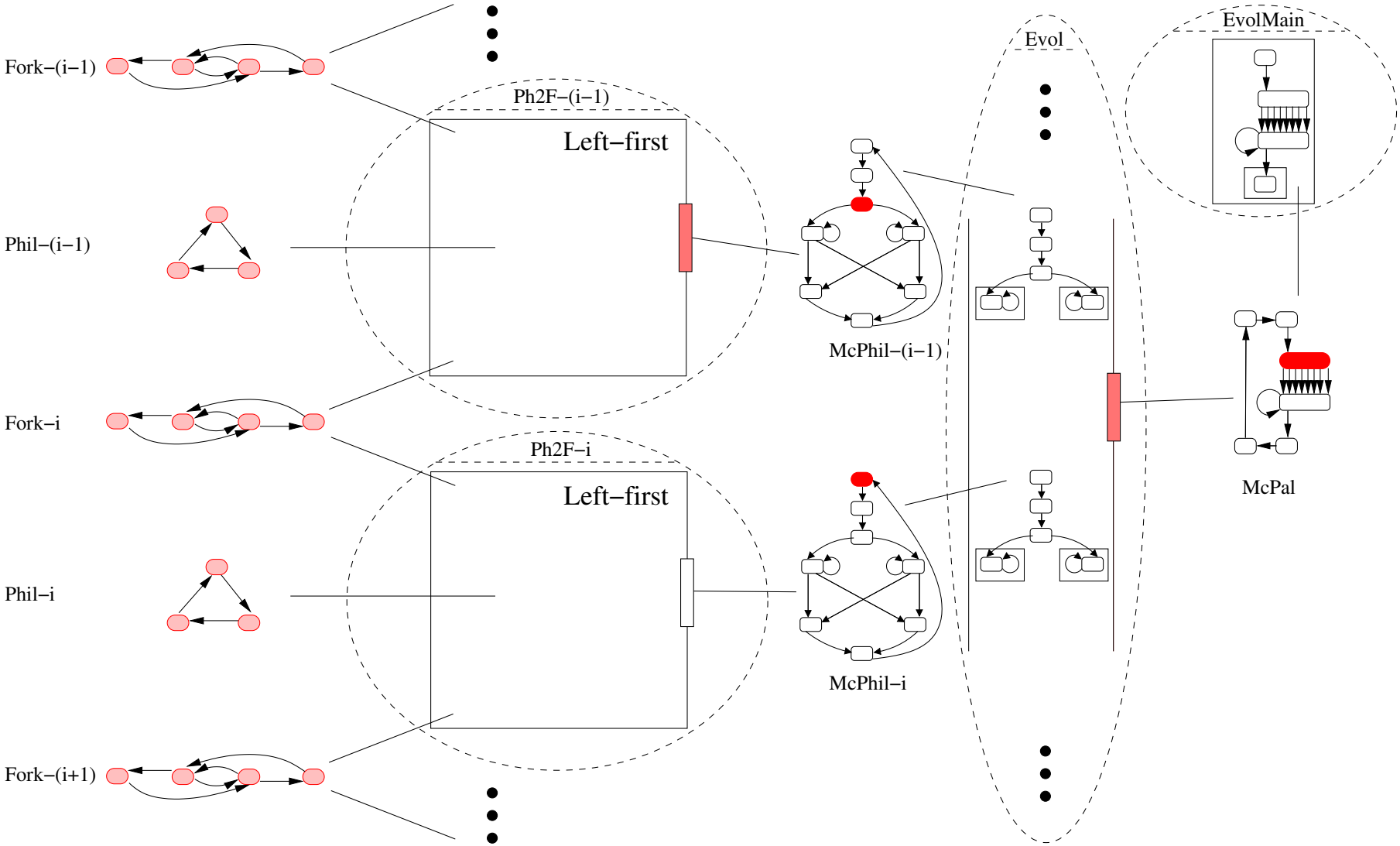
keeping

an orchestrated equivalent of ongoing as-is choreography

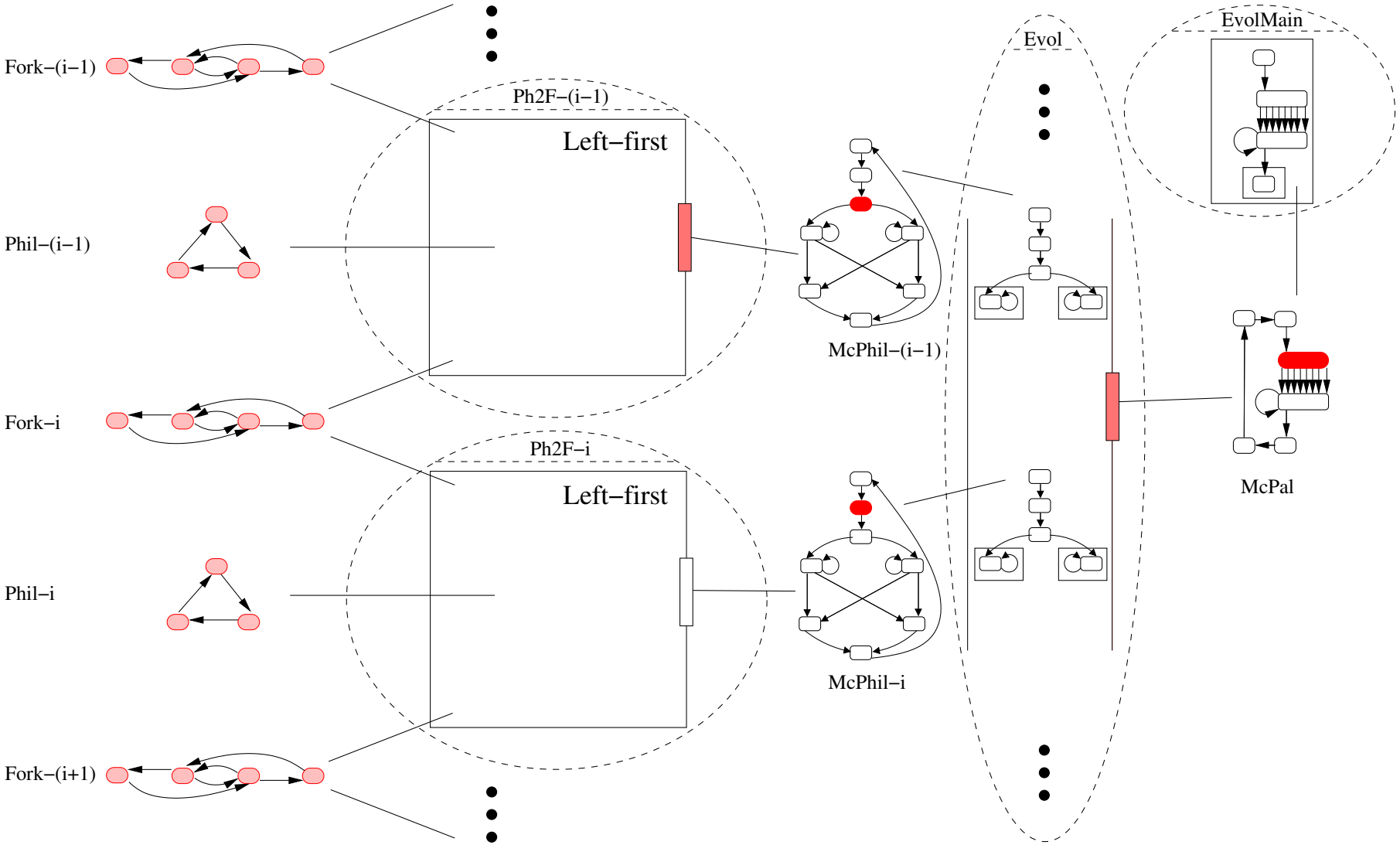
animation of migration coordination



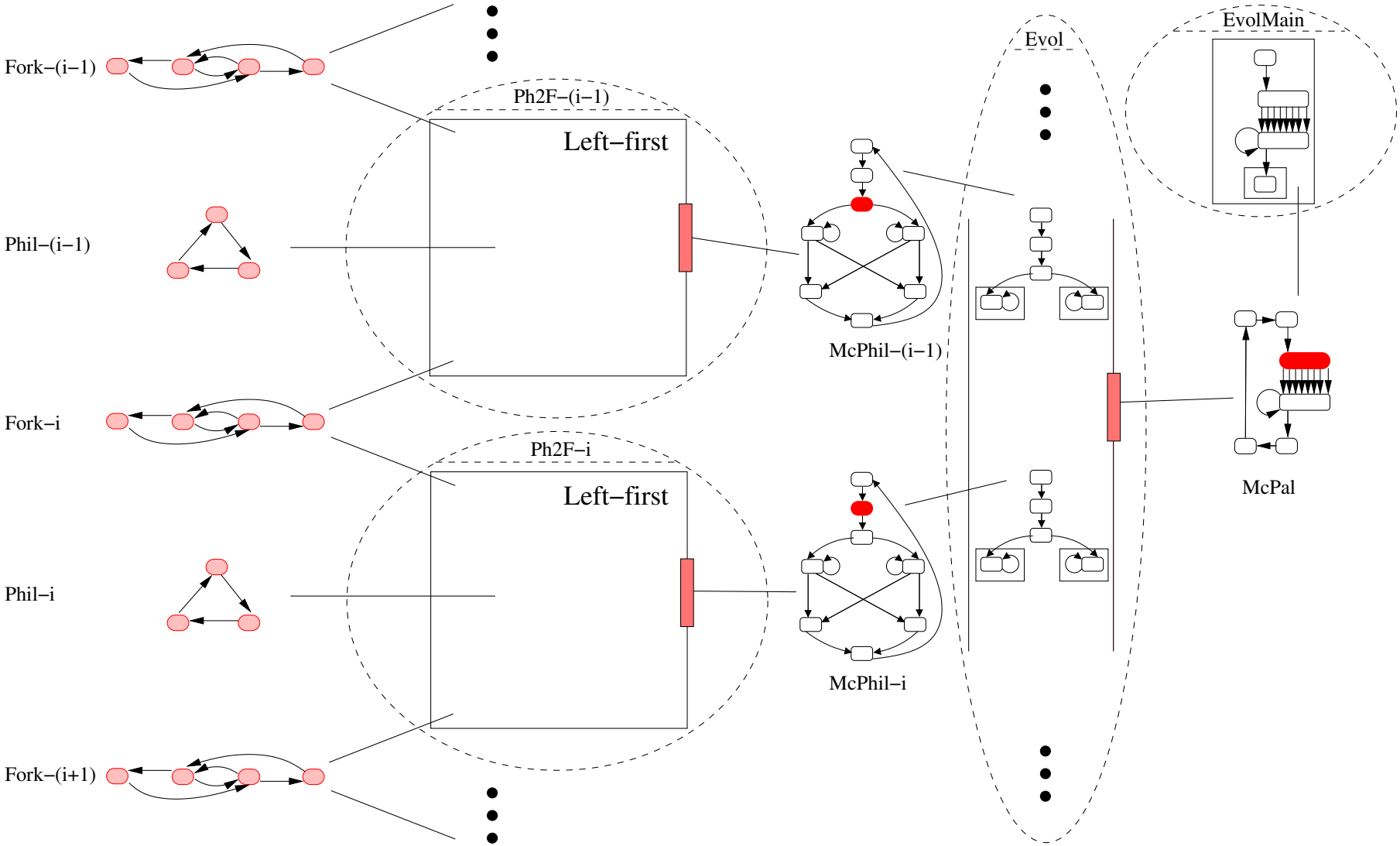
animation of migration coordination



animation of migration coordination



animation of migration coordination



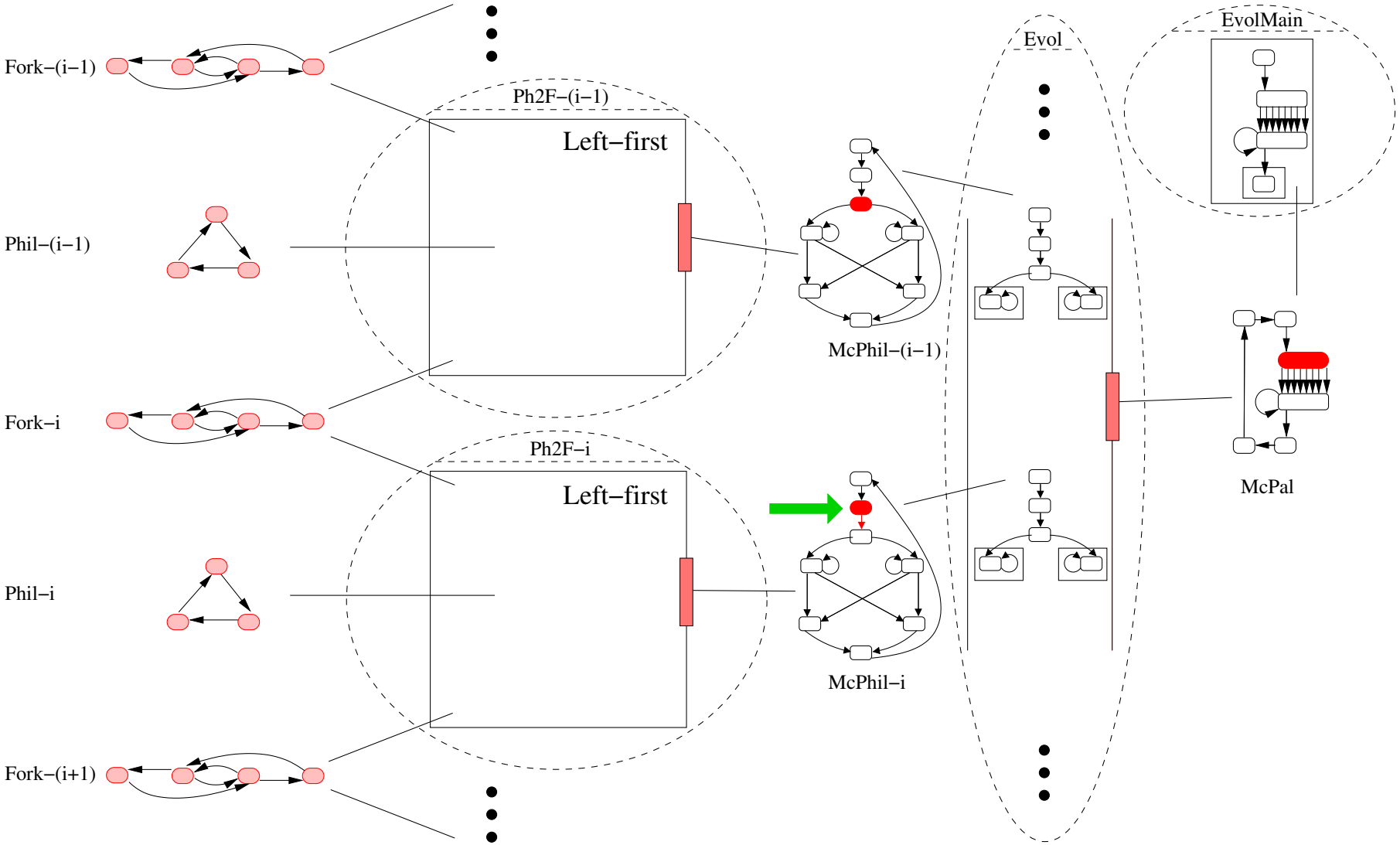
animation of migration coordination

orchestration rule

$$McPhil_i : Awake \xrightarrow{\text{takeOver}} JoiningIn * McPhil_i : [Crs_i := Crs_i - Crs_{i,asls}]$$

is now applicable

animation of migration coordination



animation of migration coordination

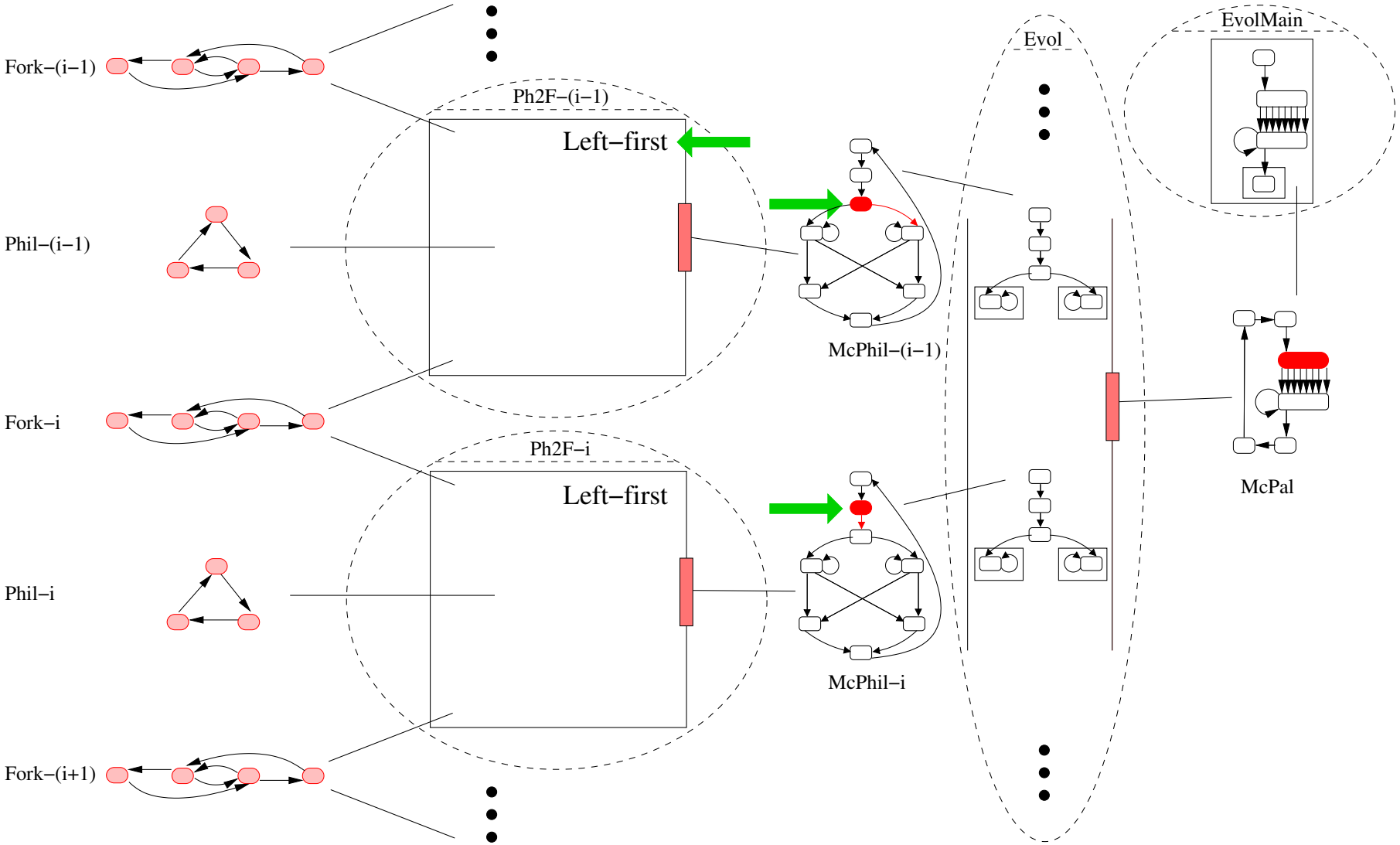
and orchestration rule

$$McPhil_{i-1} : JoiningIn \xrightarrow{\text{conductToR}} ToR \ * \ldots$$

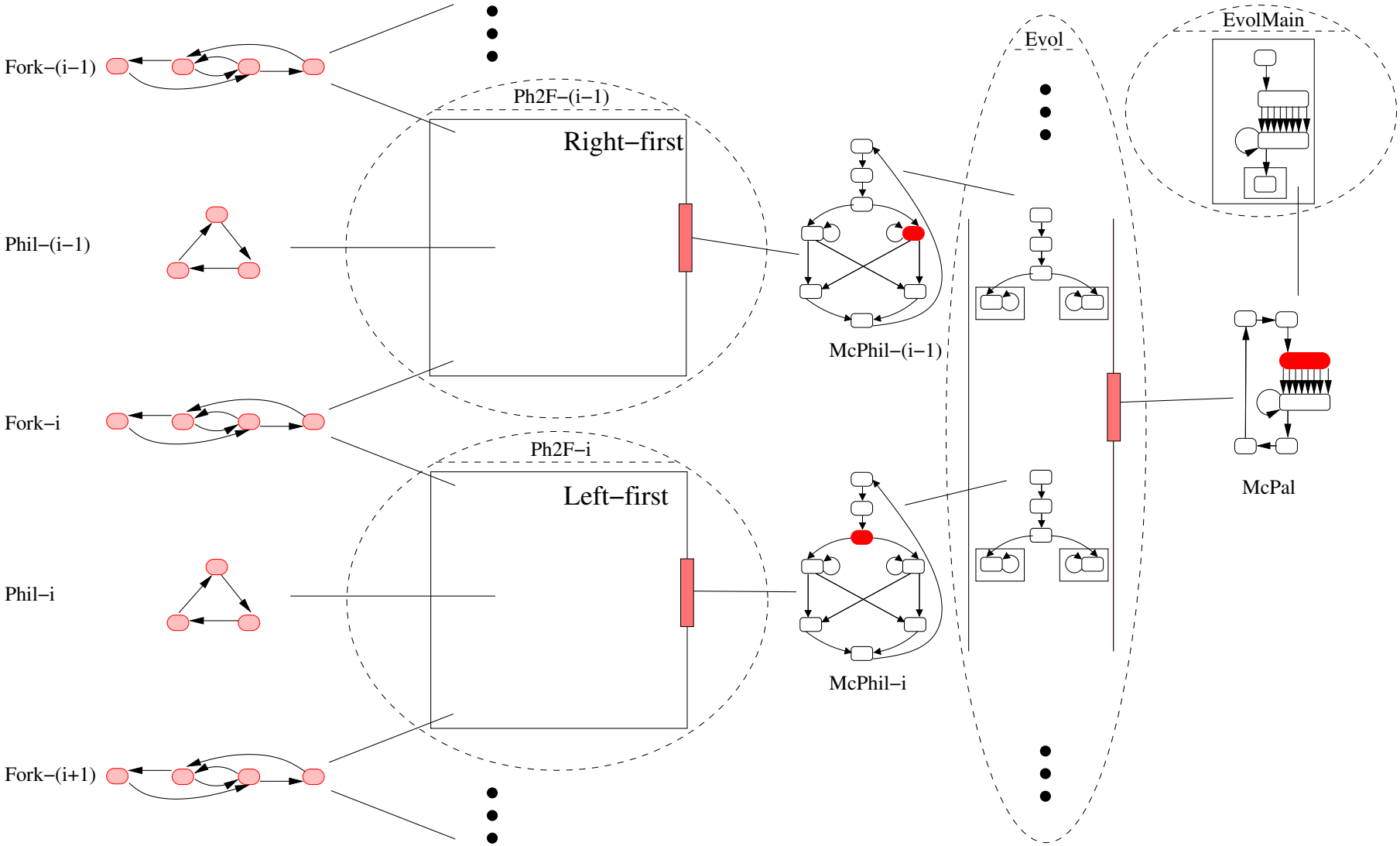
$$\ldots \qquad 1 \ out \ of \ 3 \ cases$$

is now applicable too

animation of migration coordination



animation of migration coordination



animation of migration coordination

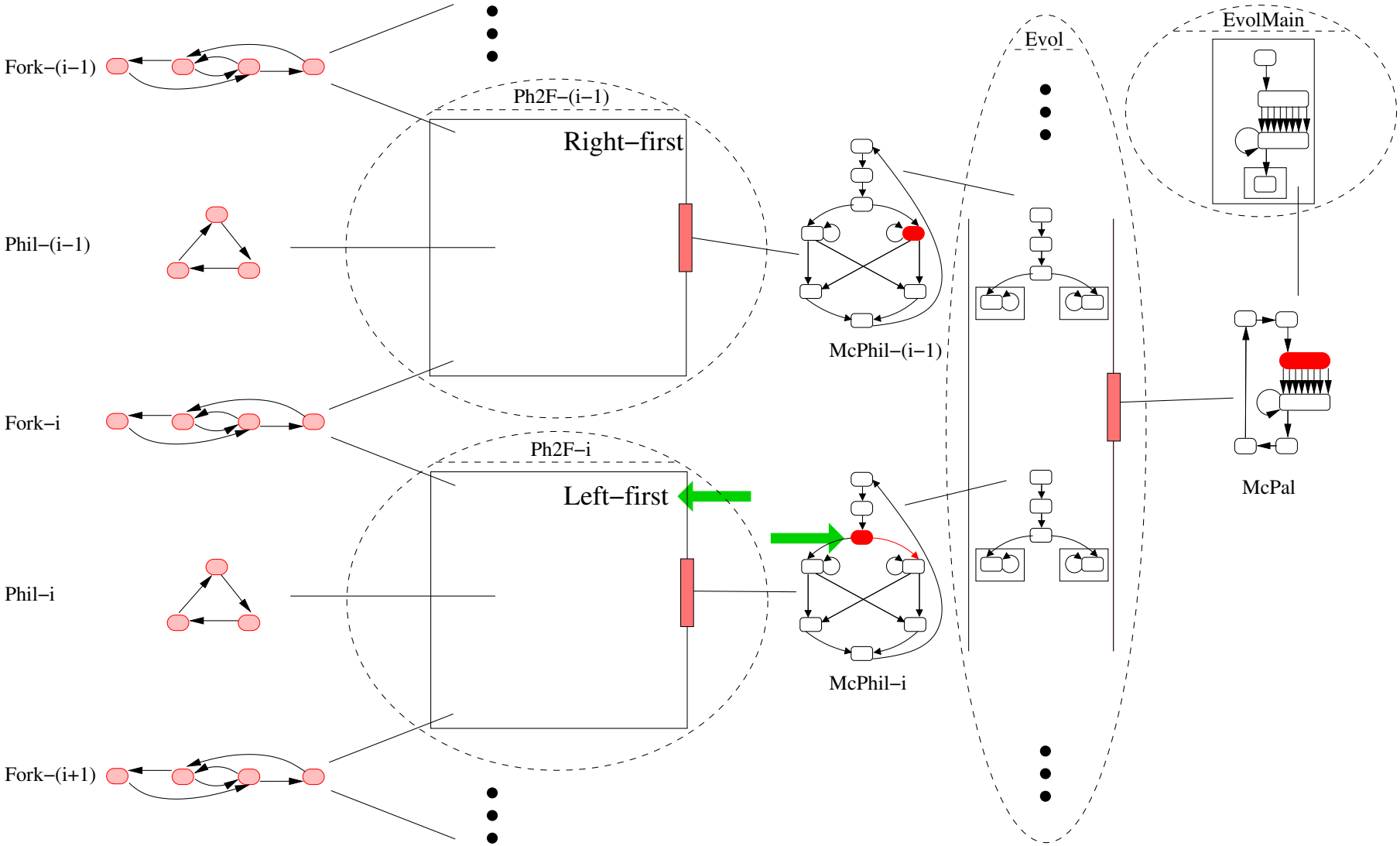
orchestration rule

$McPhil_i : JoiningIn \xrightarrow{\text{conductToR}} ToR \; * \ldots$

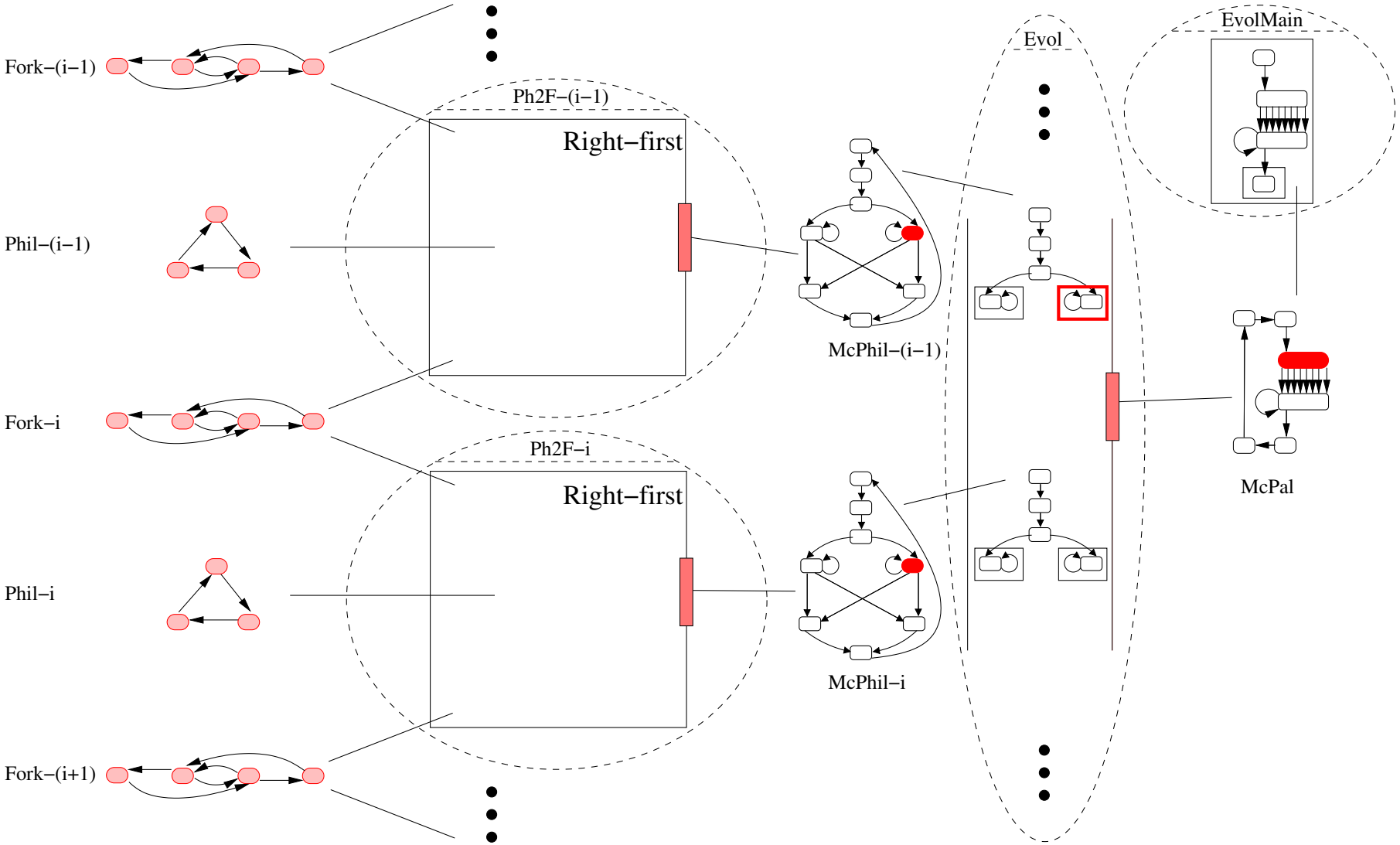
$\ldots \qquad 1 \; out \; of \; 3 \; cases$

is now applicable

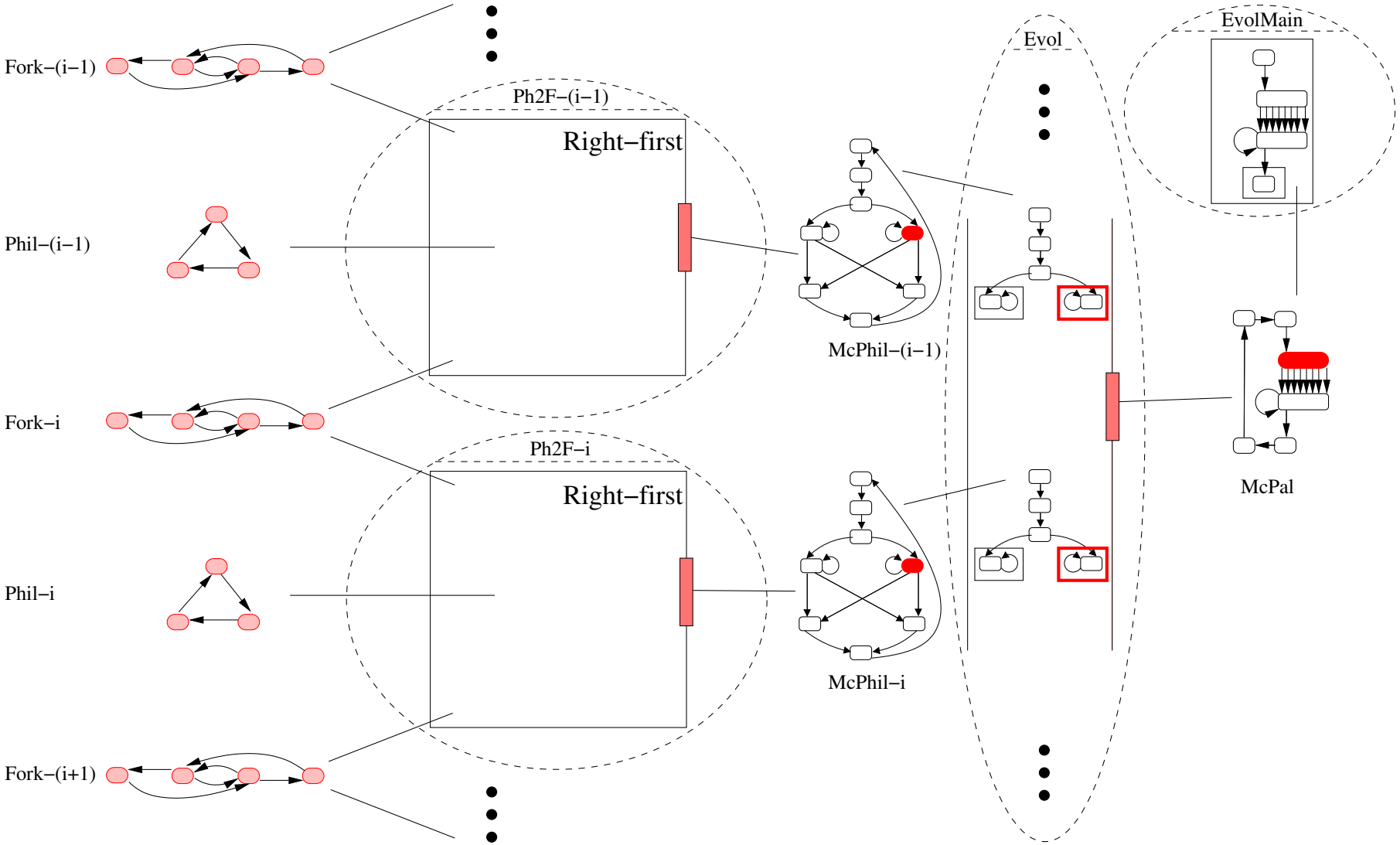
animation of migration coordination



animation of migration coordination



animation of migration coordination



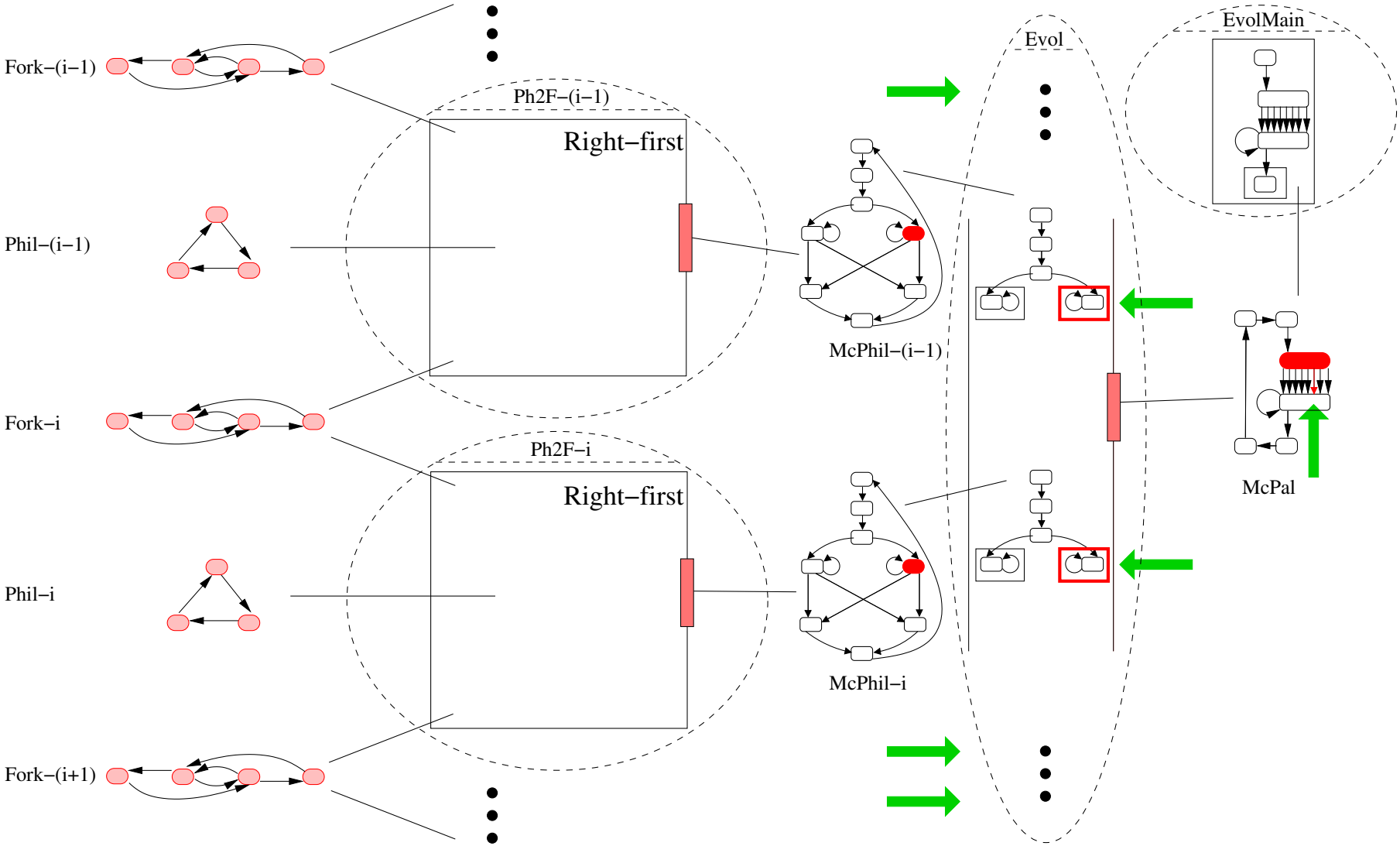
animation of migration coordination

eventually, orchestration rule

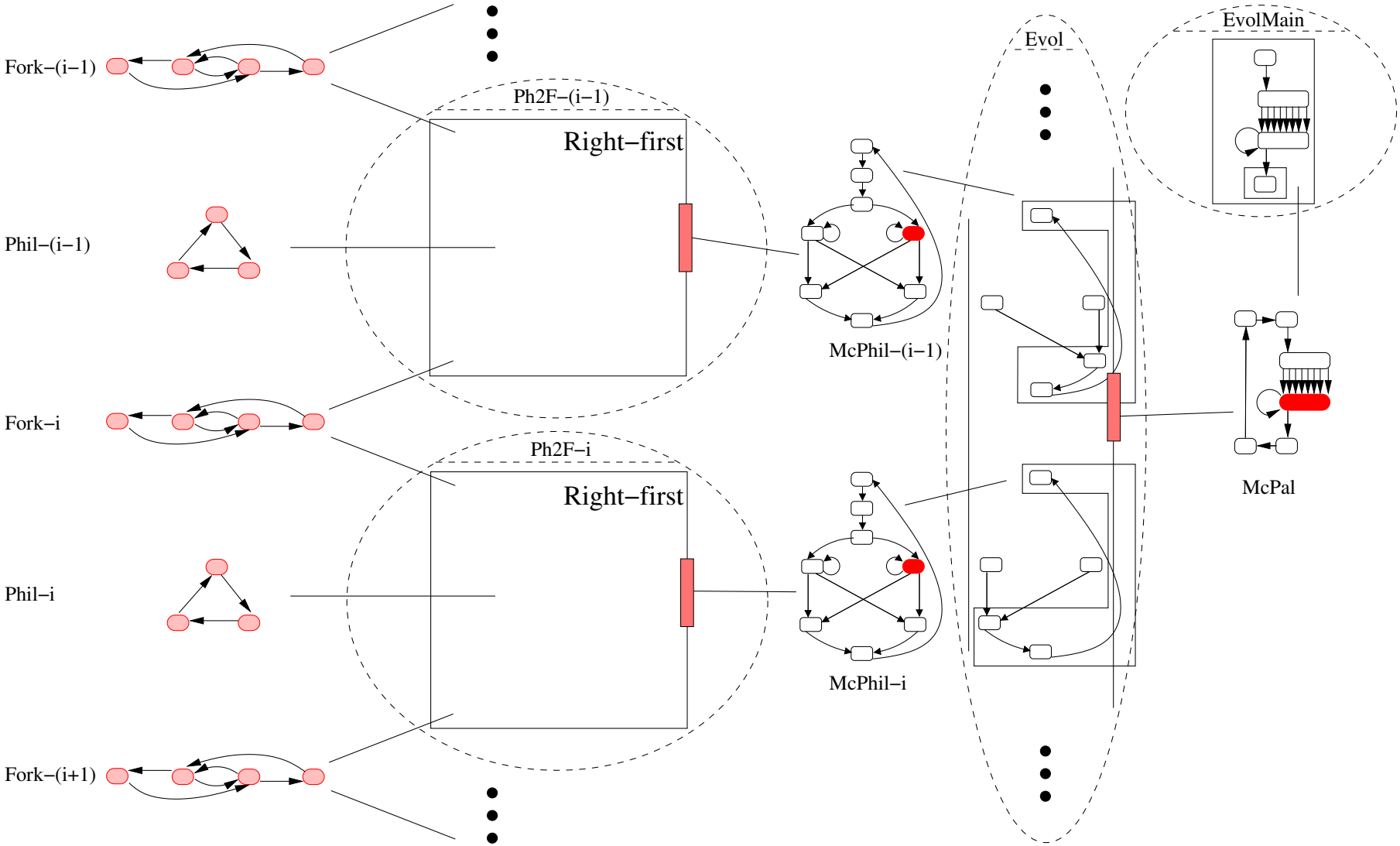
$$McPal : Delegated \xrightarrow{\text{singleSwapRL}} Gathering *$$
$$McPhil_j(Evol) : Active \xrightarrow{\text{halfwayR}} EndAsR,$$
$$McPhil_{j+1}(Evol) : Active \xrightarrow{\text{halfwayR}} EndAsL,$$
$$McPhil_{j+2}(Evol) : Active \xrightarrow{\text{halfwayL}} EndAsR,$$
$$McPhil_{j+3}(Evol) : Active \xrightarrow{\text{halfwayL}} EndAsL,$$
$$McPhil_{j+4}(Evol) : Active \xrightarrow{\text{halfwayL}} EndAsL$$

is applicable (for $j = i - 1$)

animation of migration coordination



animation of migration coordination



animation of migration coordination

eventually orchestration rule

$McPhil_i : ToR \overset{\text{choreofyToL}}{\longrightarrow} ToBeL \ * \ldots$

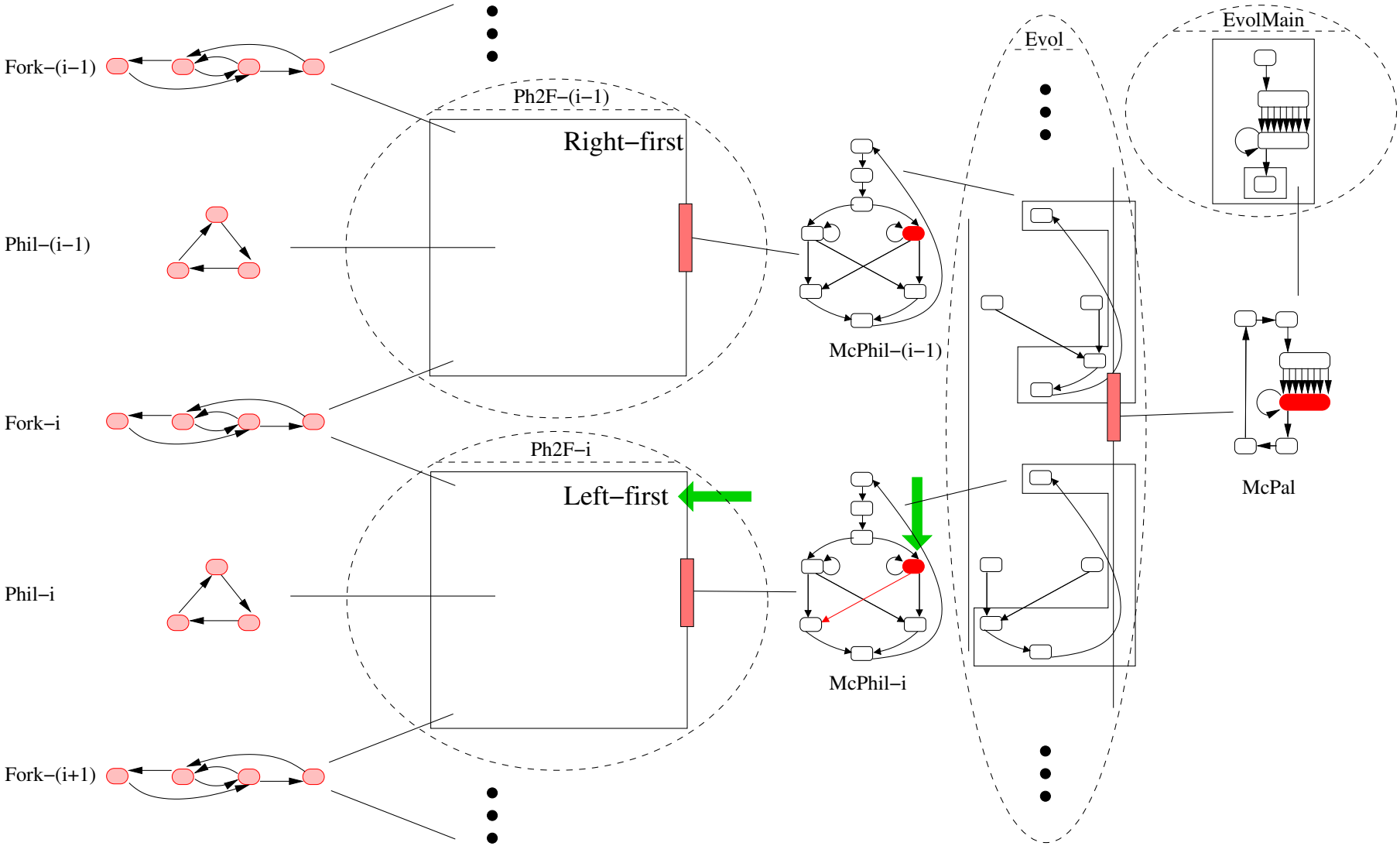
$\ldots \quad 1 \ out \ of \ 5 \ cases$

is applicable

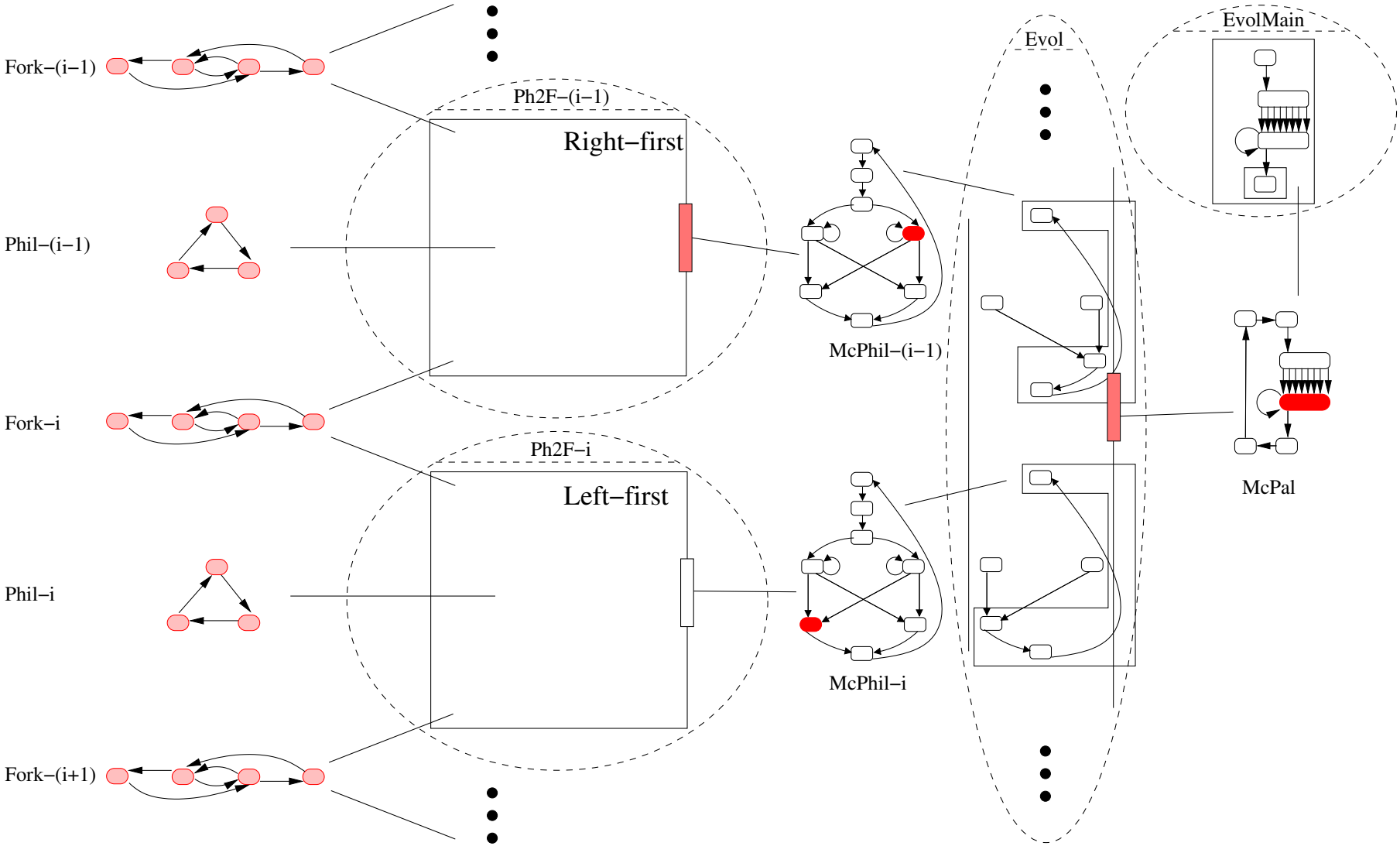
keeping

a choreographical equivalent of ongoing as-is orchestration

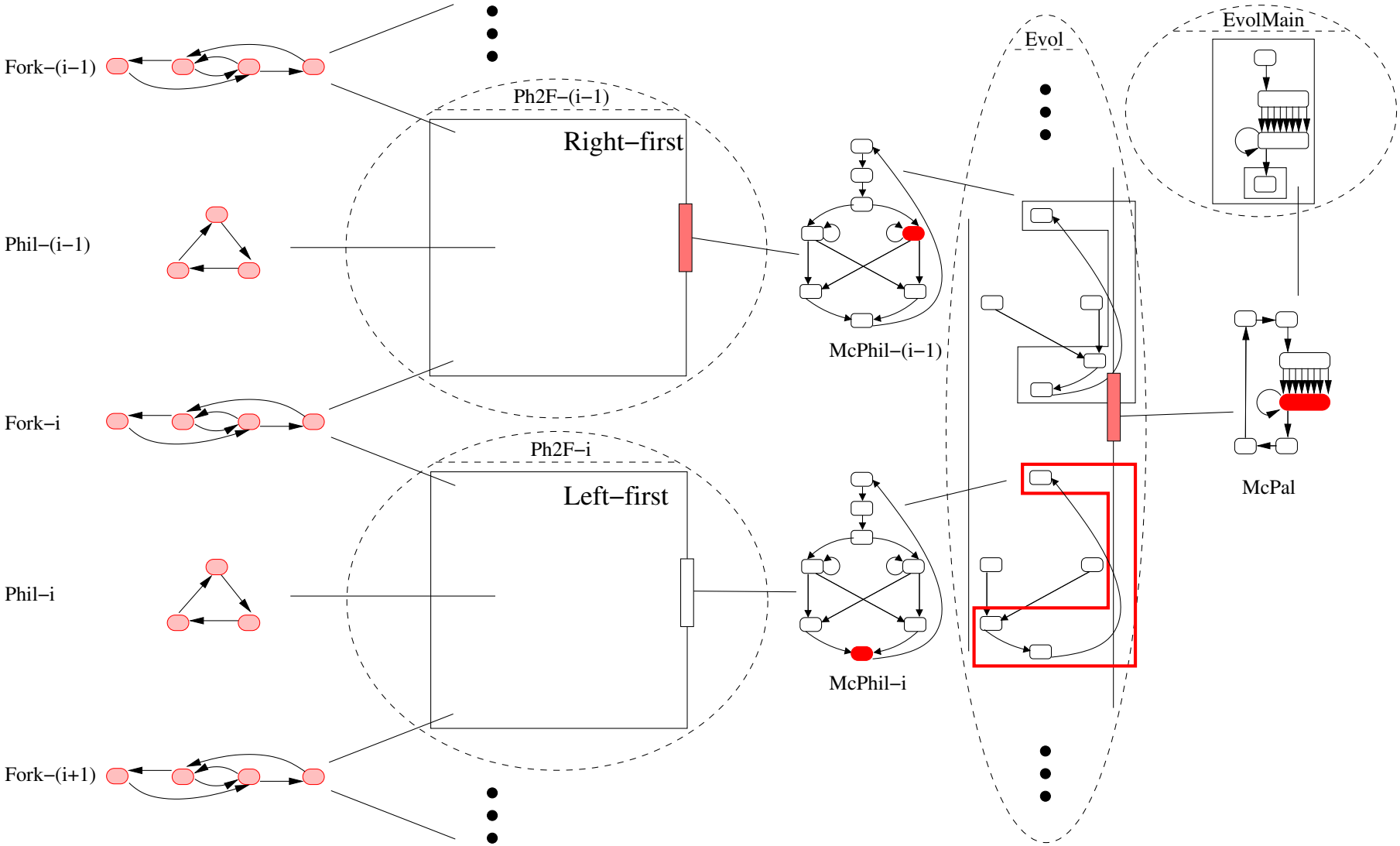
animation of migration coordination



animation of migration coordination



animation of migration coordination



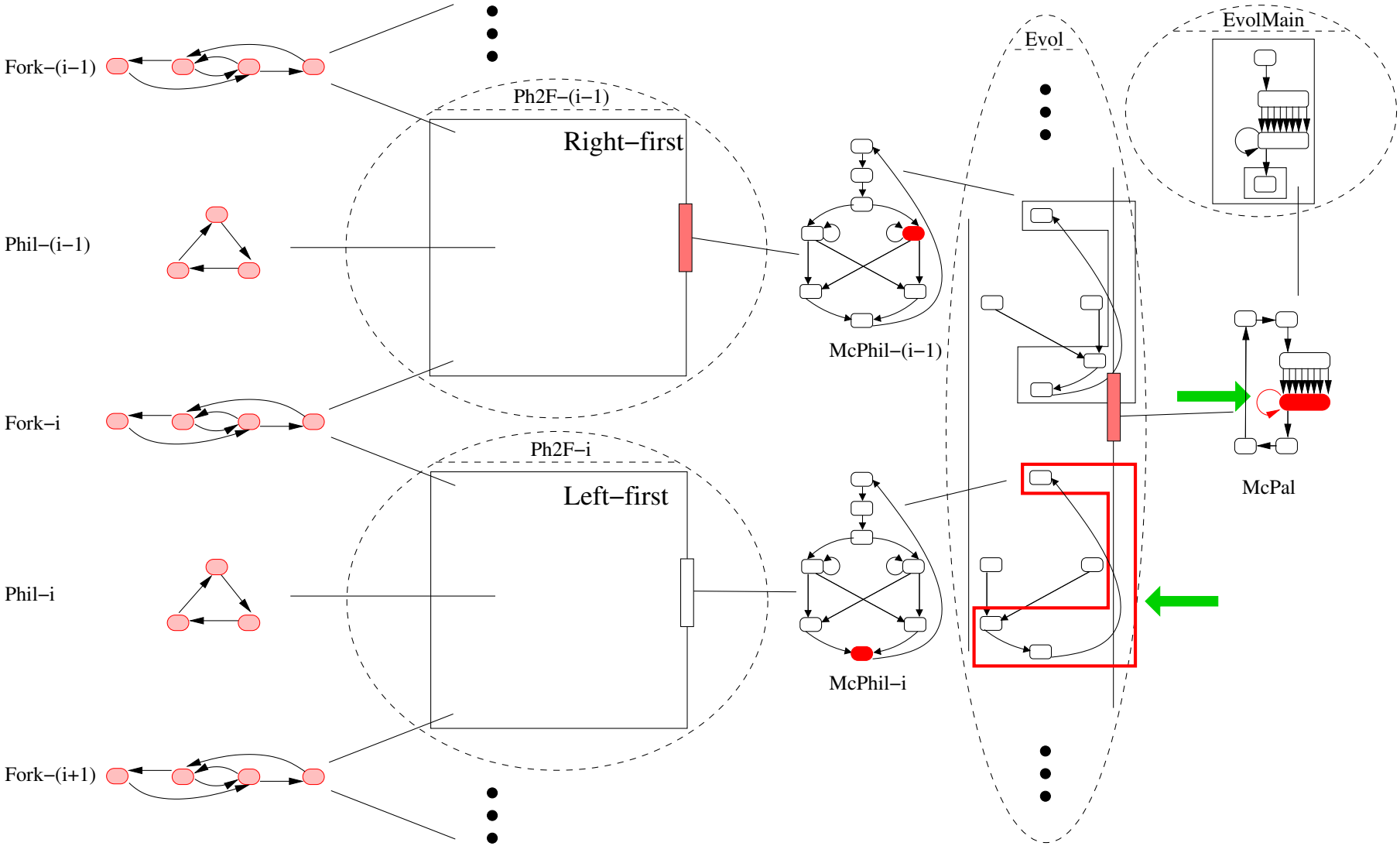
animation of migration coordination

orchestration rule

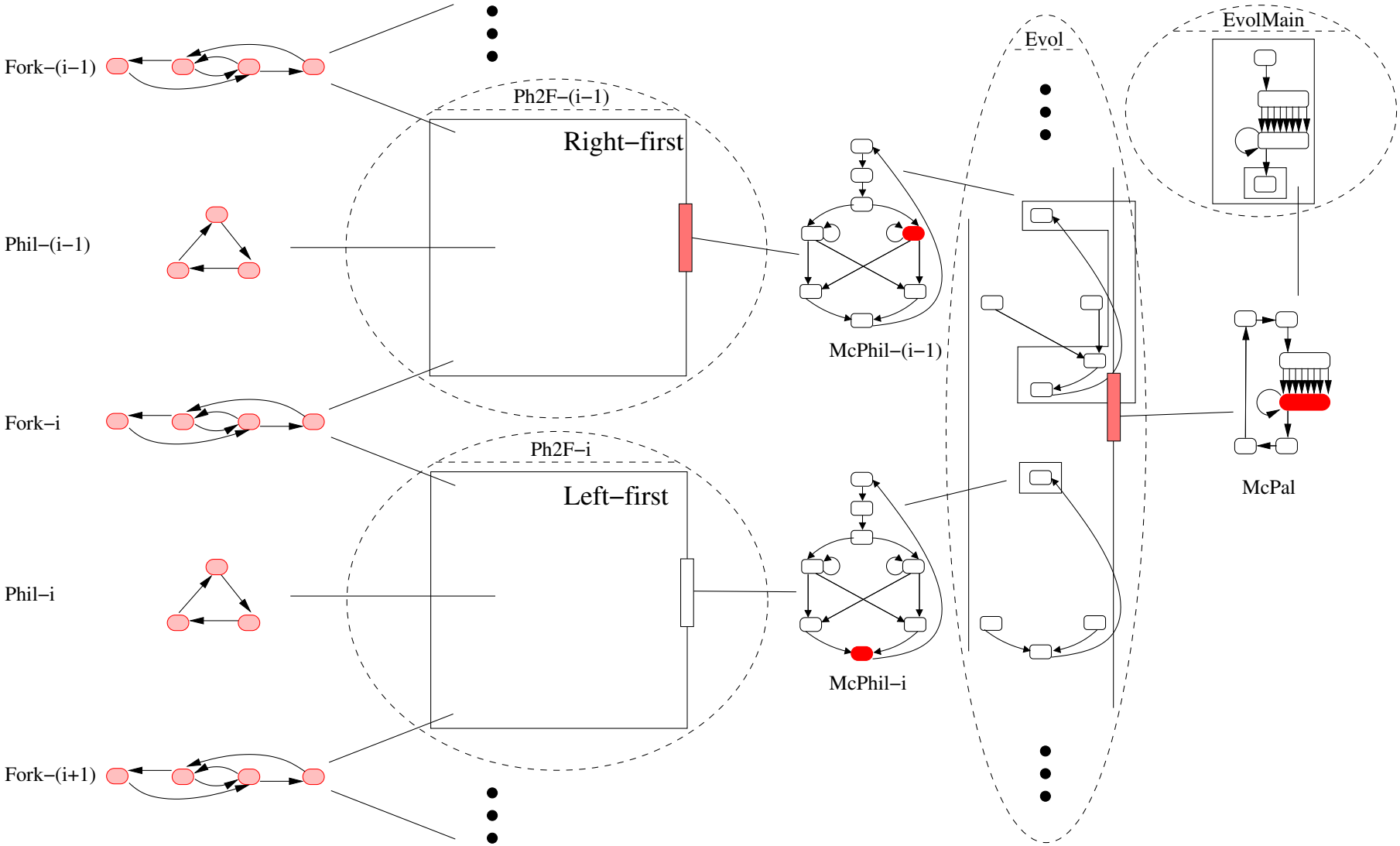
$$McPal : Gathering \xrightarrow{\text{collect}} Gathering \; *$$
$$McPhil_i(Evol) : EndAsL \xrightarrow{\text{done}} Retreating,$$
$$McPal : [Crs := Crs + Crs_{i,toBe},$$
$$Crs_{toBe} := Crs_{toBe} + Crs_{i,toBe}]$$

is now applicable

animation of migration coordination



animation of migration coordination



animation of migration coordination

eventually orchestration rule

$McPhil_{i-1} : ToR \overset{\text{choreofyToR}}{\longrightarrow} ToBeR \ * \ldots$

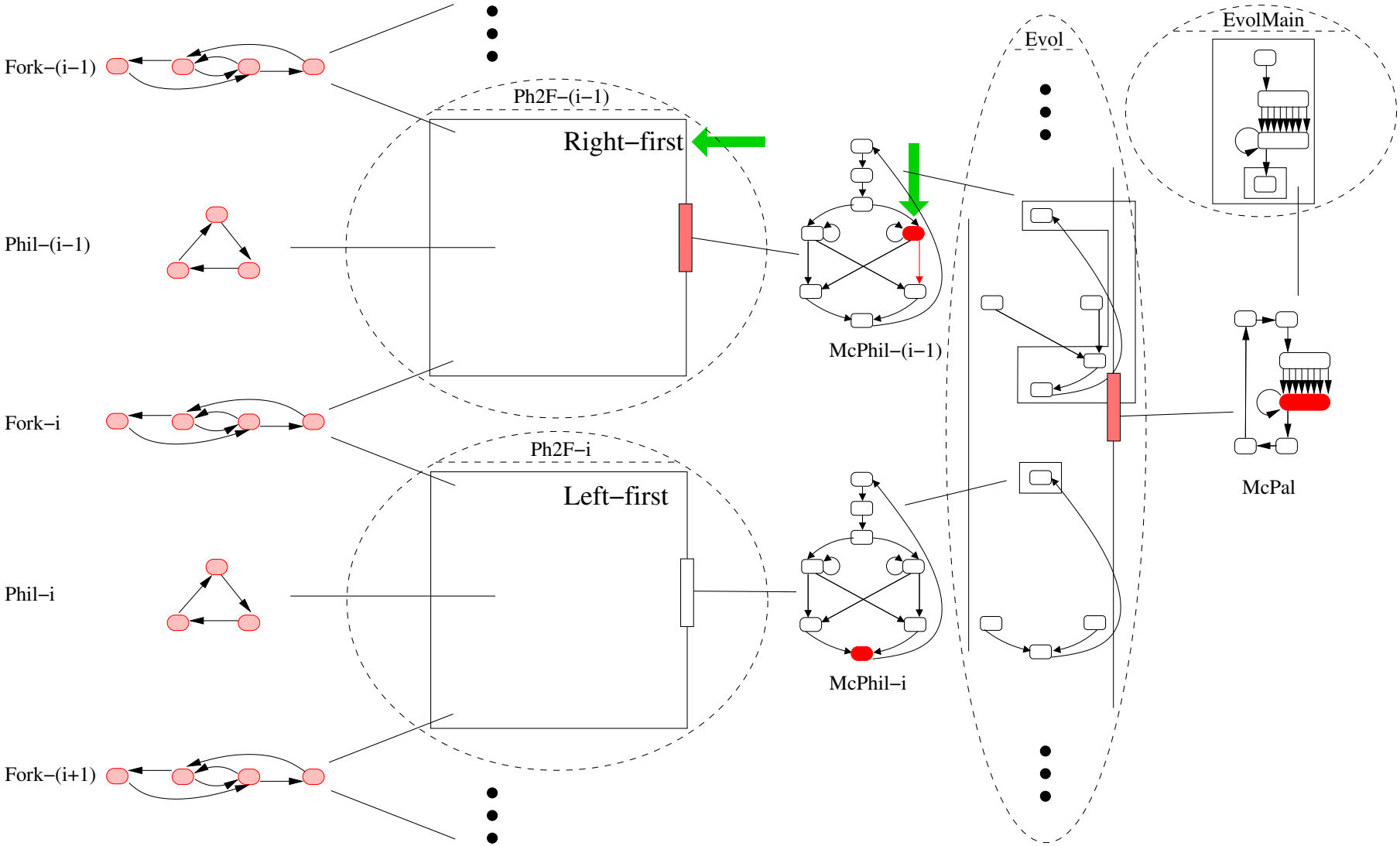
$\ldots \qquad 1 \textit{ out of } 4 \textit{ cases}$

is applicable

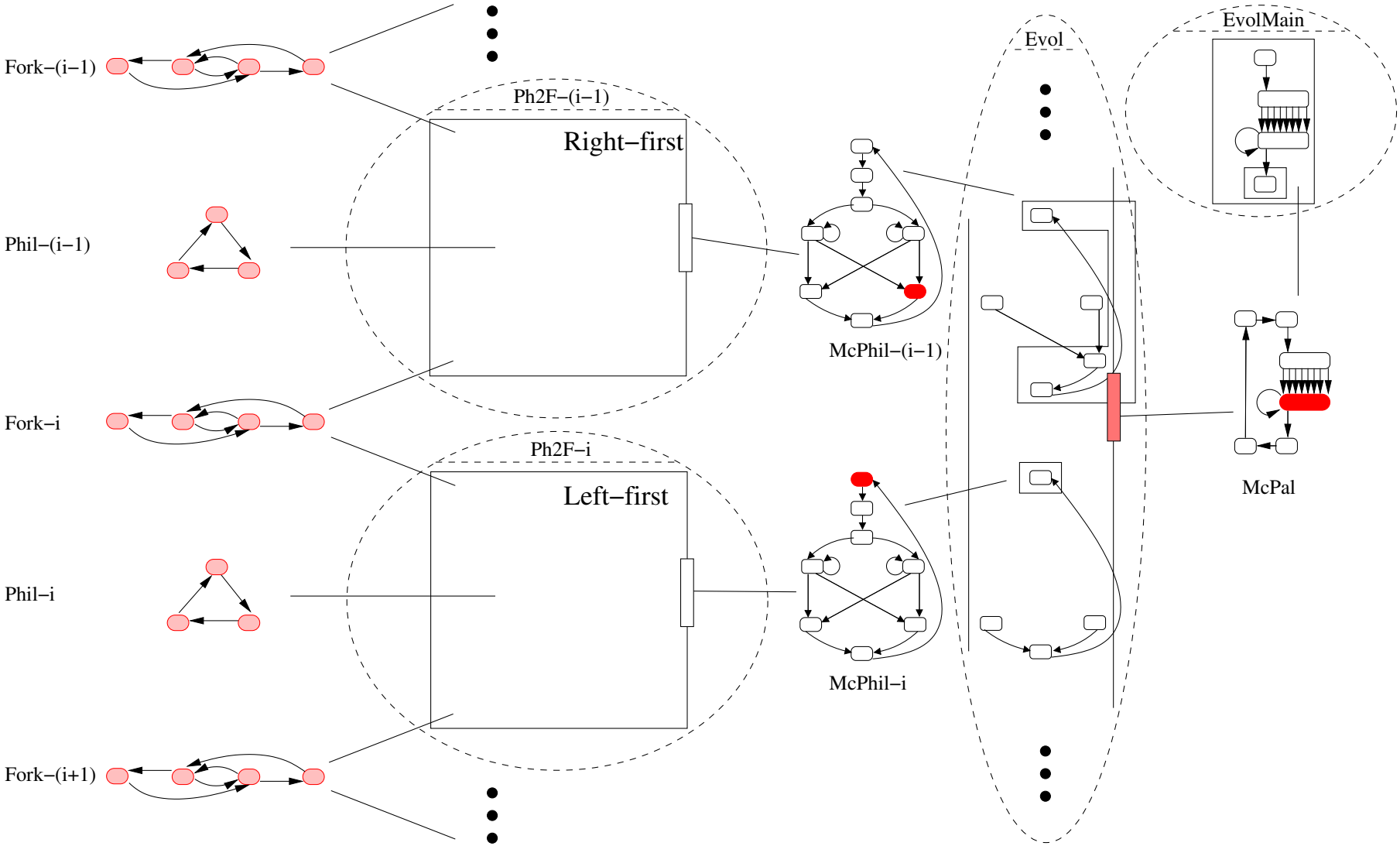
keeping

a choreographical equivalent of ongoing as-is orchestration

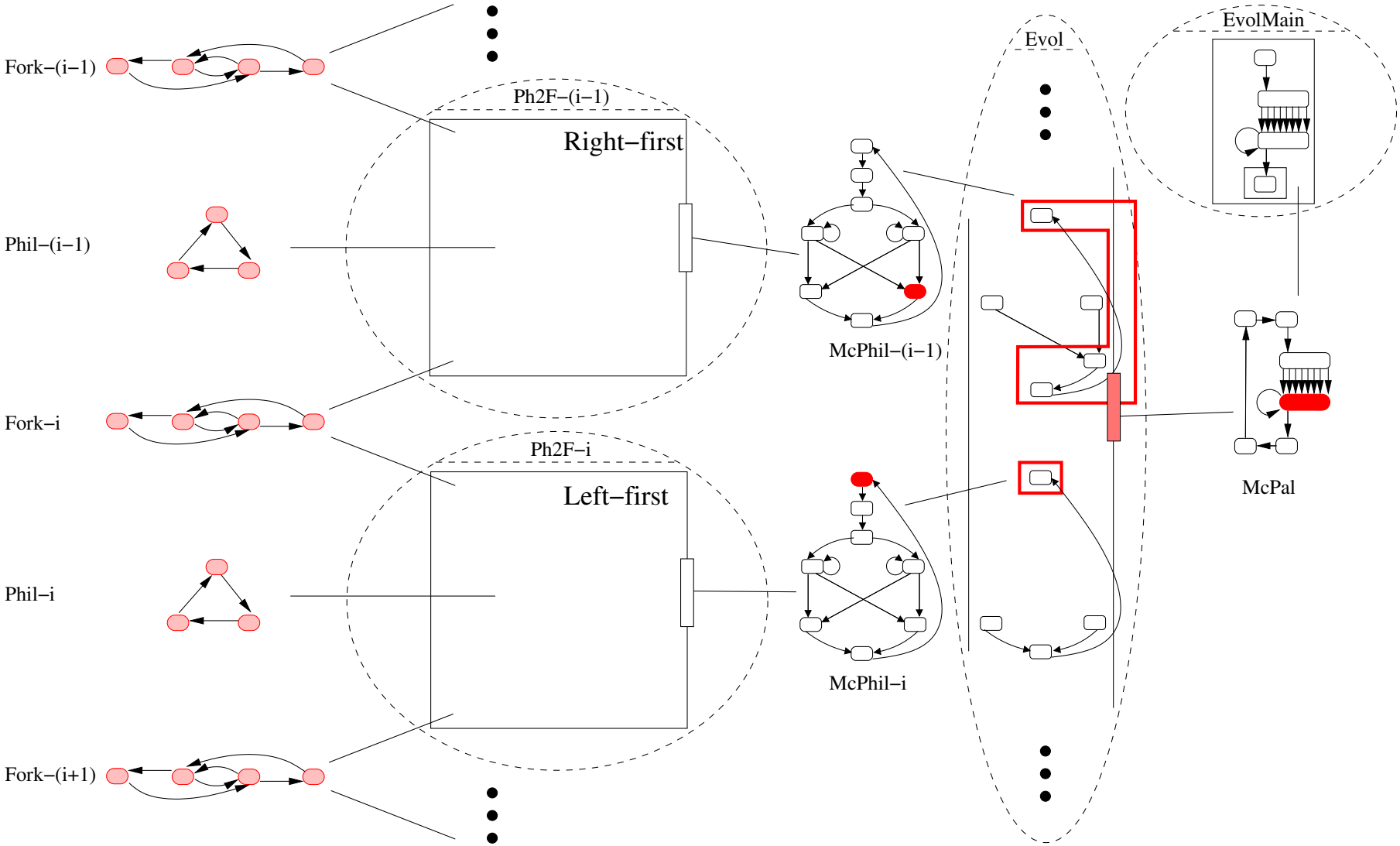
animation of migration coordination



animation of migration coordination



animation of migration coordination



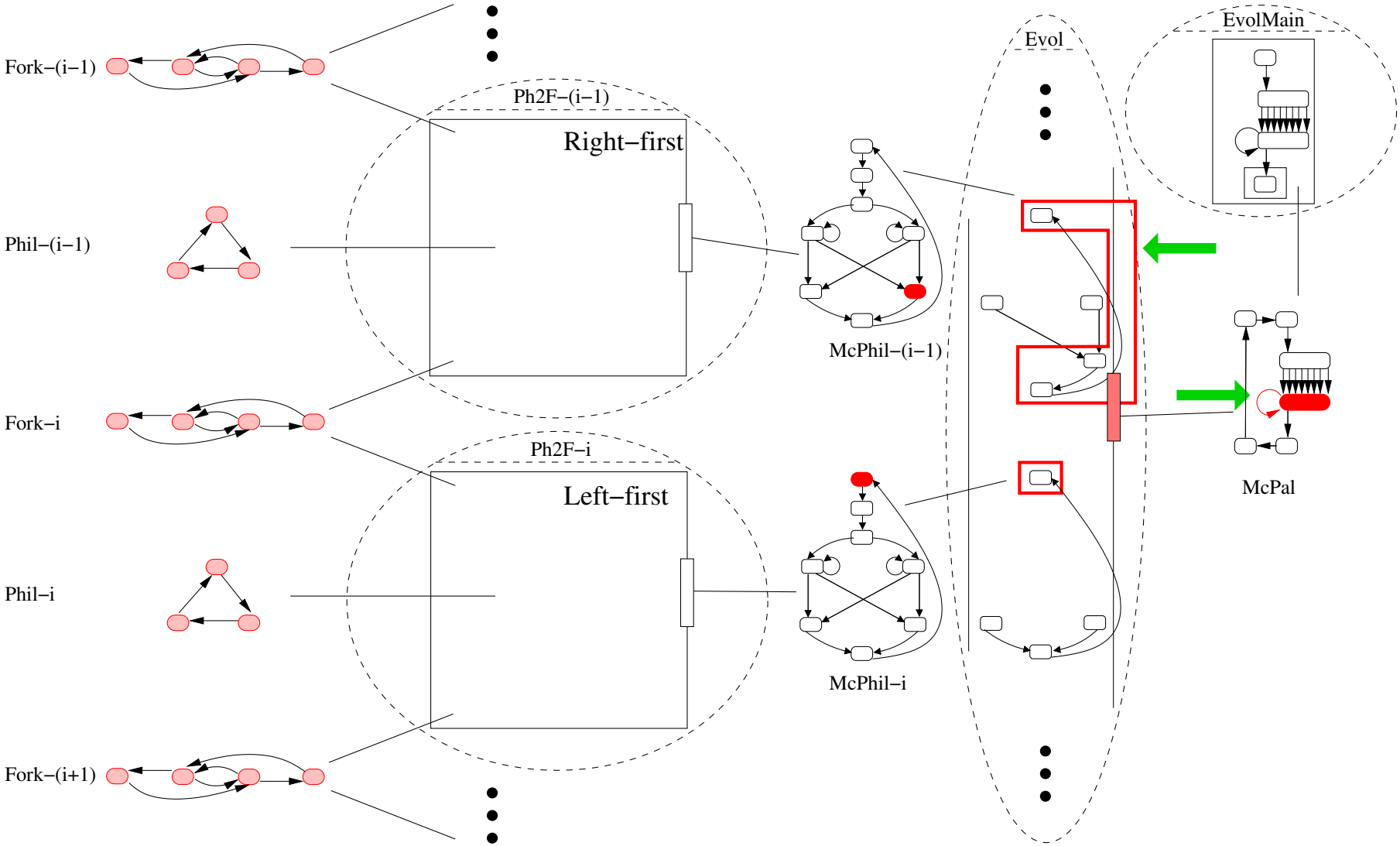
animation of migration coordination

orchestration rule

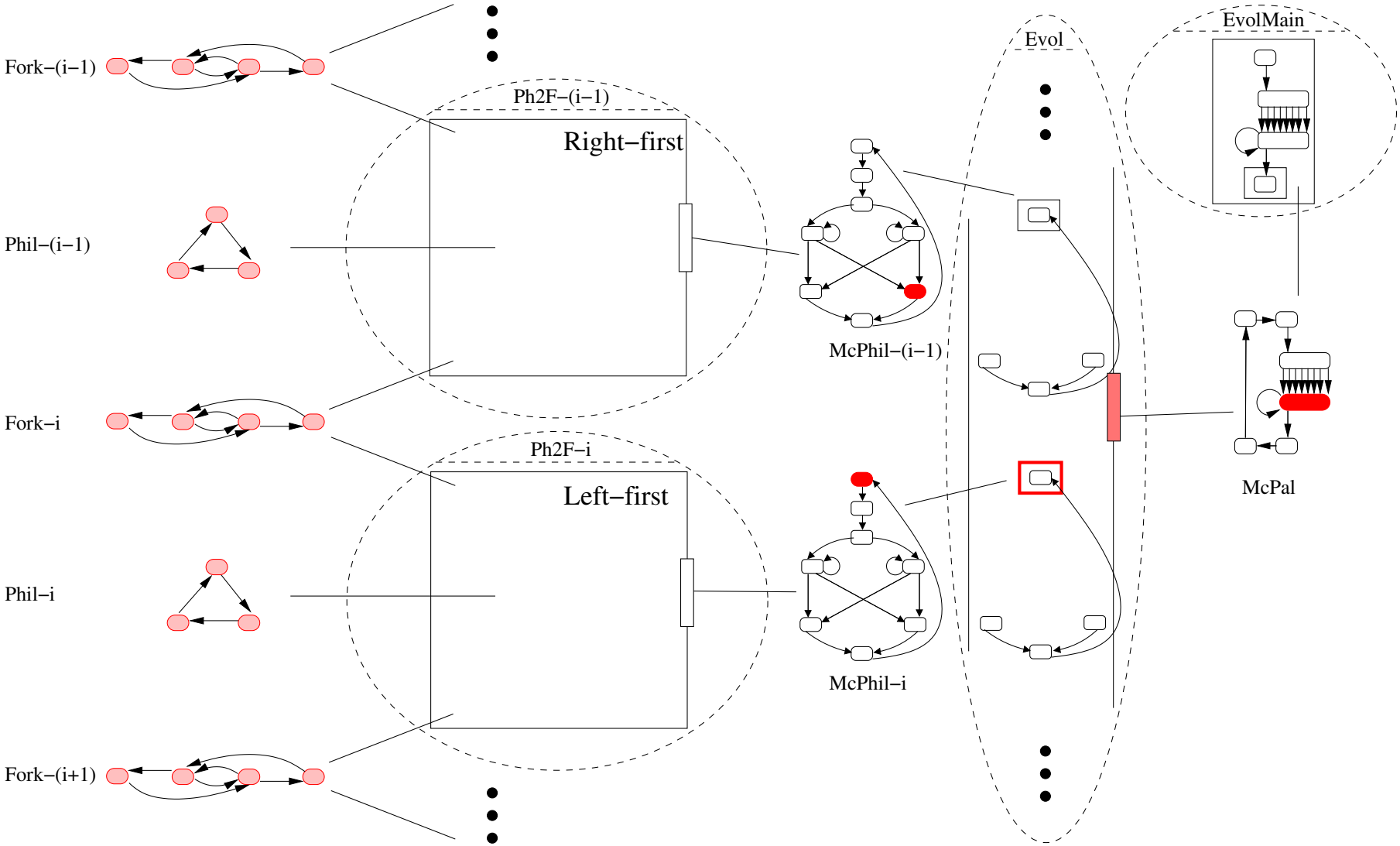
$$McPal : Gathering \xrightarrow{\text{collect}} Gathering \; * \\ McPhil_{i-1}(Evol) : EndAsL \xrightarrow{\text{done}} Retreating, \\ McPal : [Crs := Crs + Crs_{i-1,toBe}, \\ Crs_{toBe} := Crs_{toBe} + Crs_{i-1,toBe}]$$

is now applicable

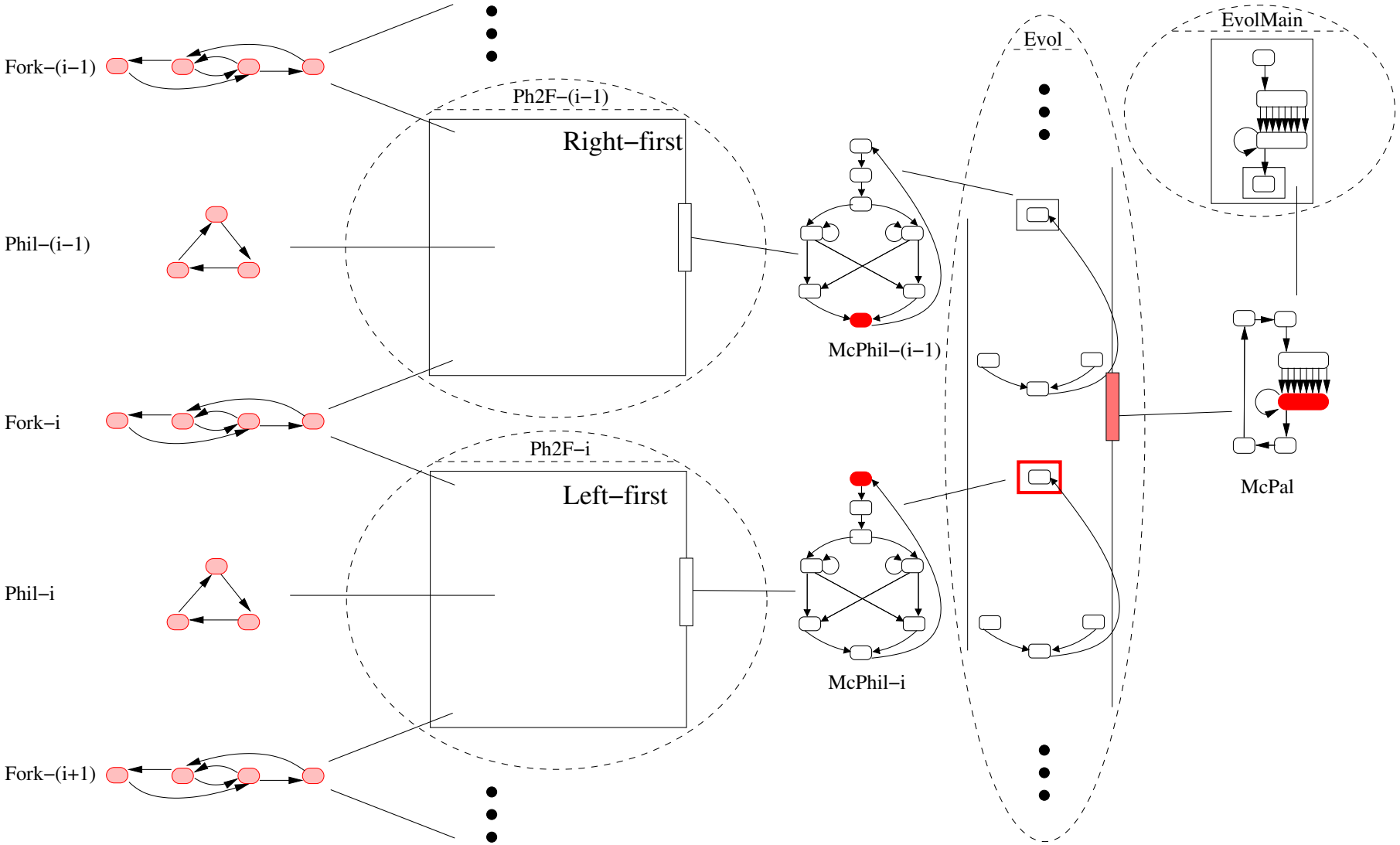
animation of migration coordination



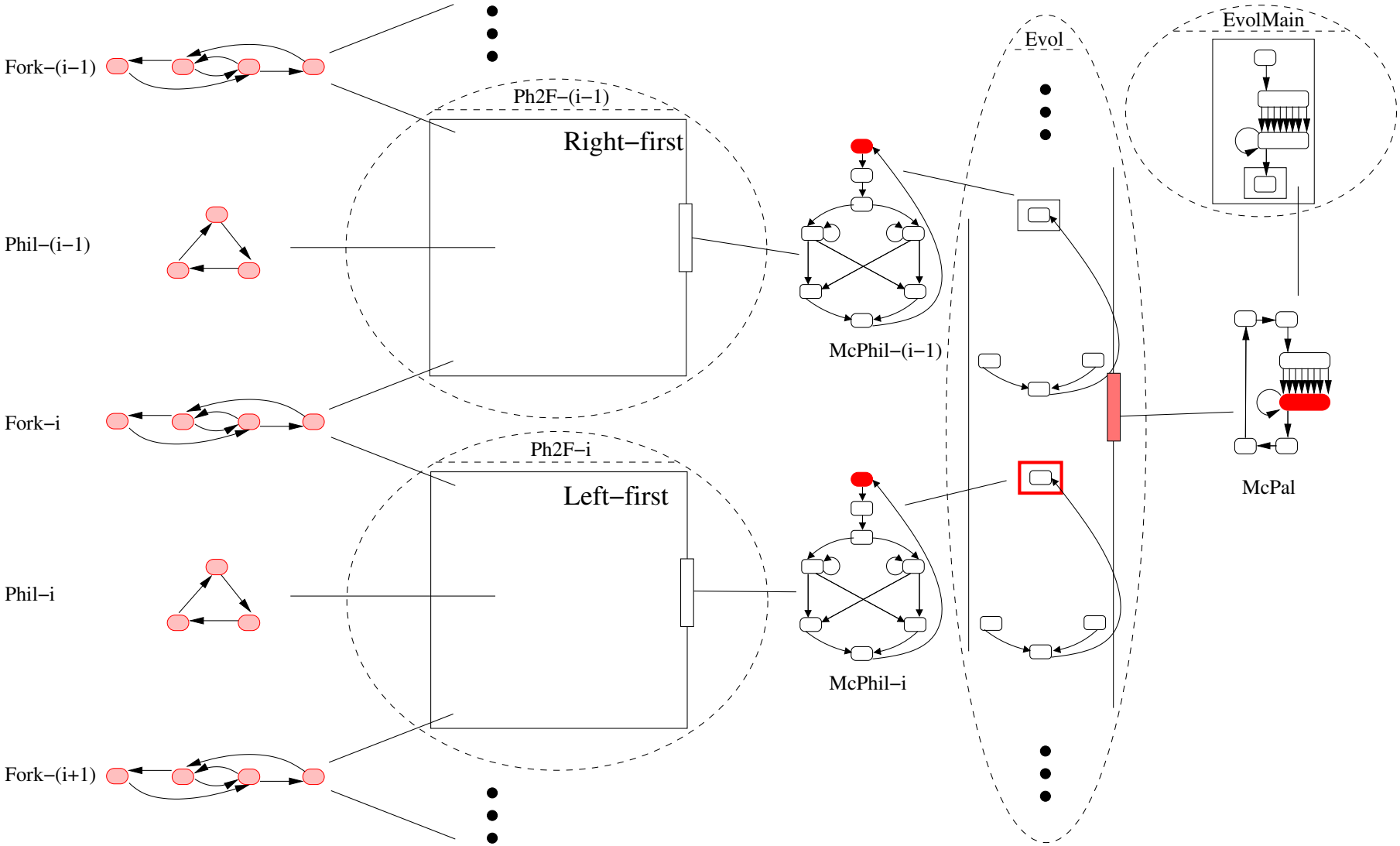
animation of migration coordination



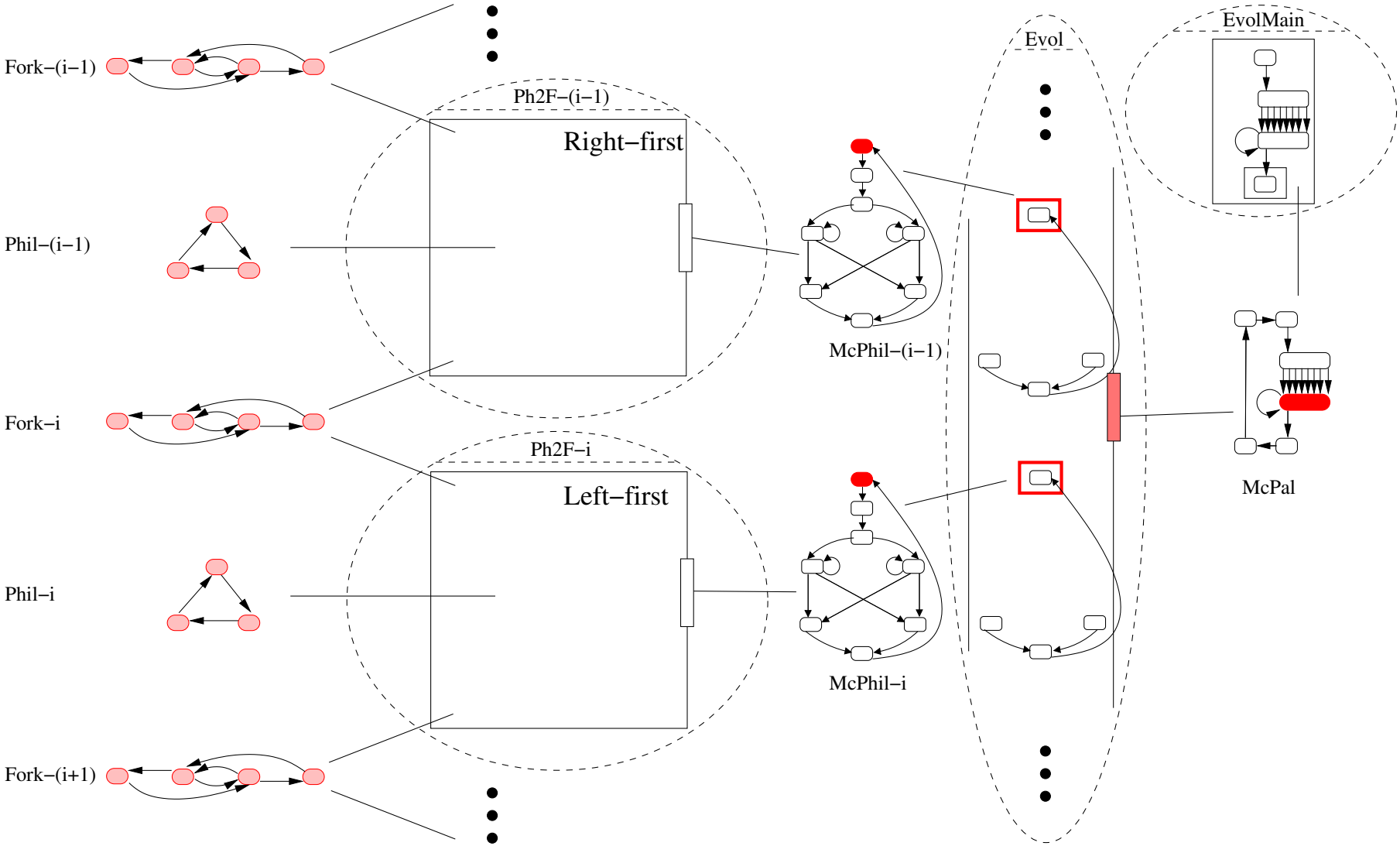
animation of migration coordination



animation of migration coordination



animation of migration coordination



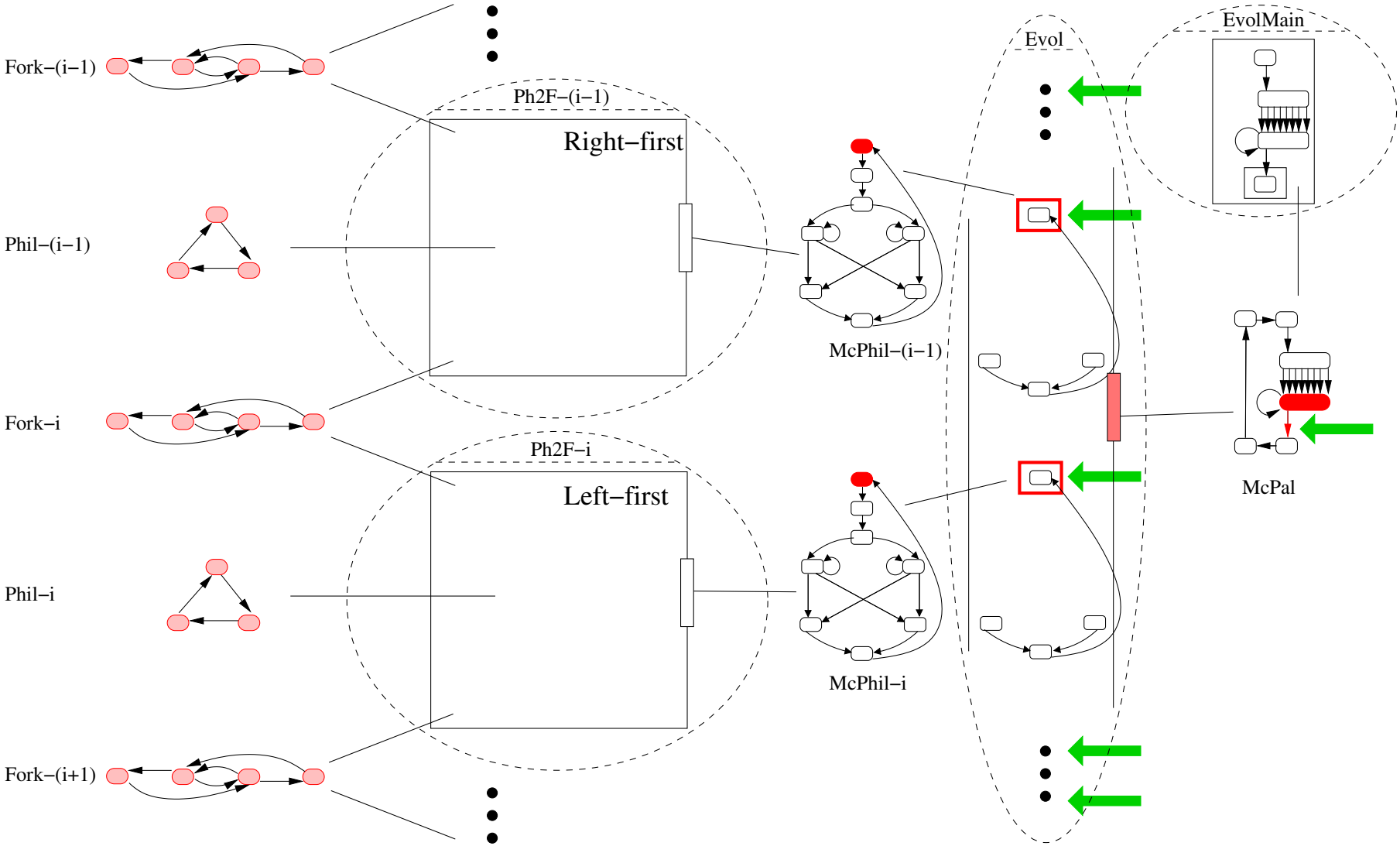
animation of migration coordination

eventually, orchestration rule

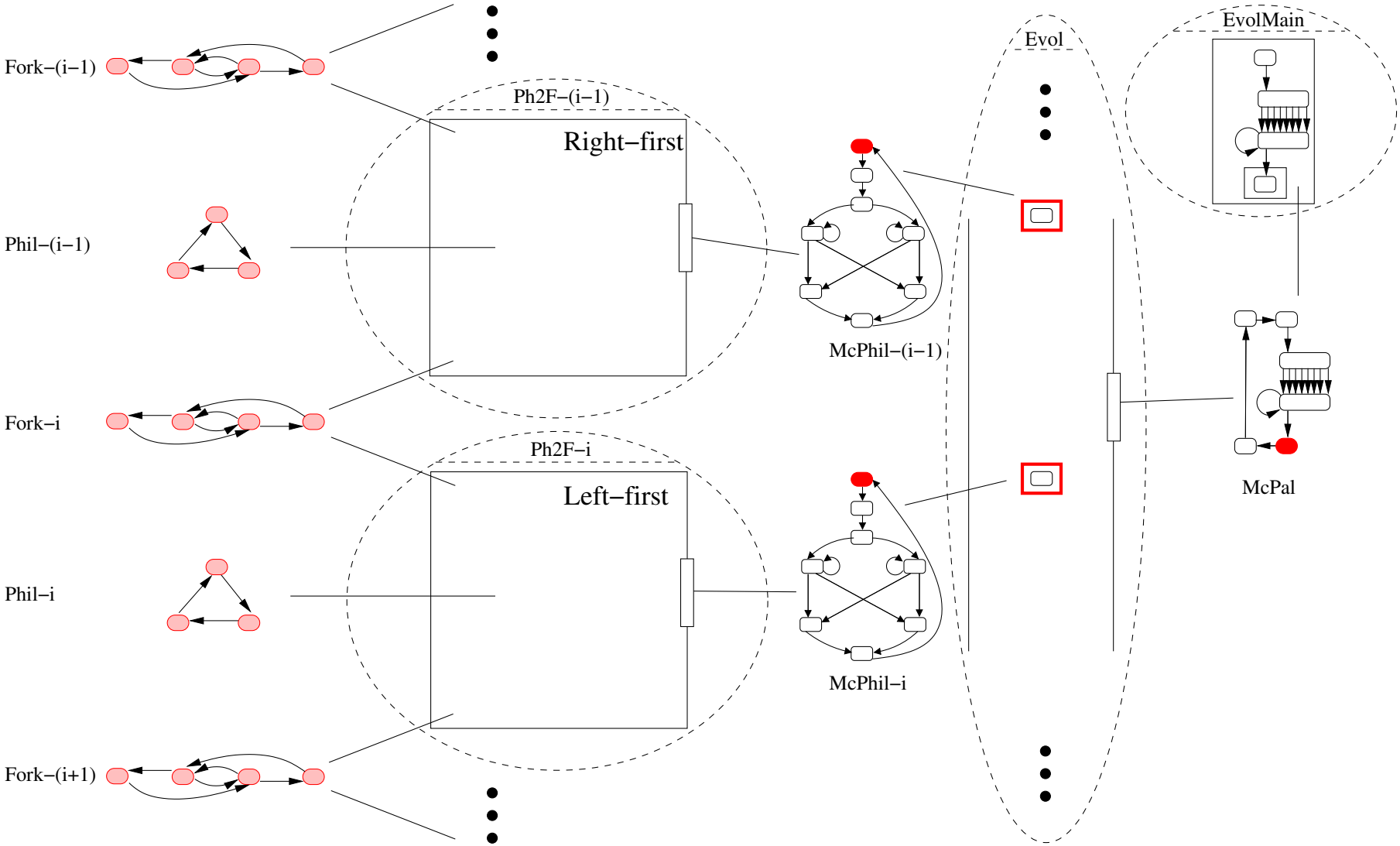
$$McPal : Gathering \xrightarrow{\text{close}} Content *$$
$$McPhil_1(Evol) : Retreating \xrightarrow{\text{away}} Passive,$$
$$McPhil_2(Evol) : Retreating \xrightarrow{\text{away}} Passive,$$
$$McPhil_3(Evol) : Retreating \xrightarrow{\text{away}} Passive,$$
$$McPhil_4(Evol) : Retreating \xrightarrow{\text{away}} Passive,$$
$$McPhil_5(Evol) : Retreating \xrightarrow{\text{away}} Passive$$

is applicable

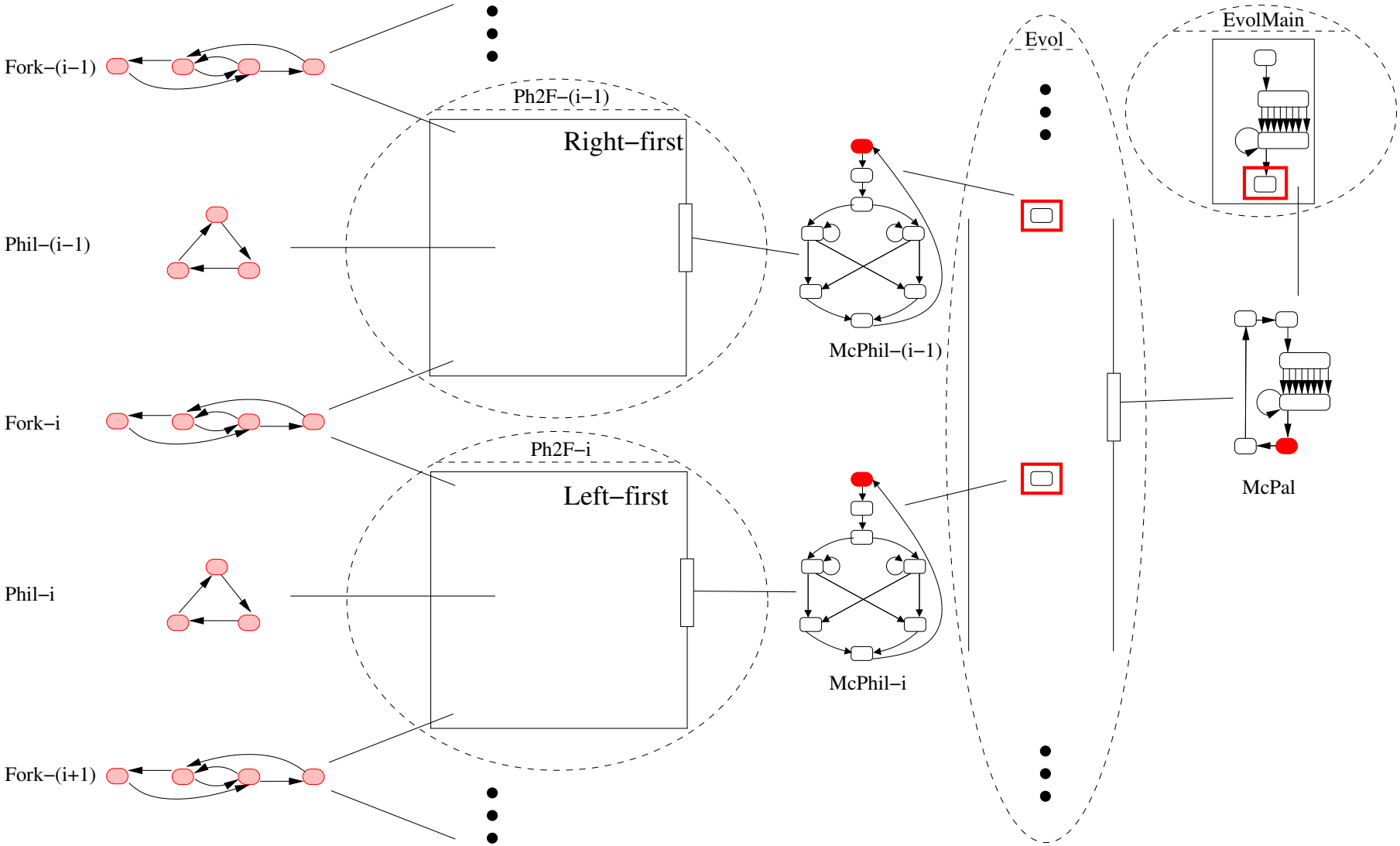
animation of migration coordination



animation of migration coordination



animation of migration coordination



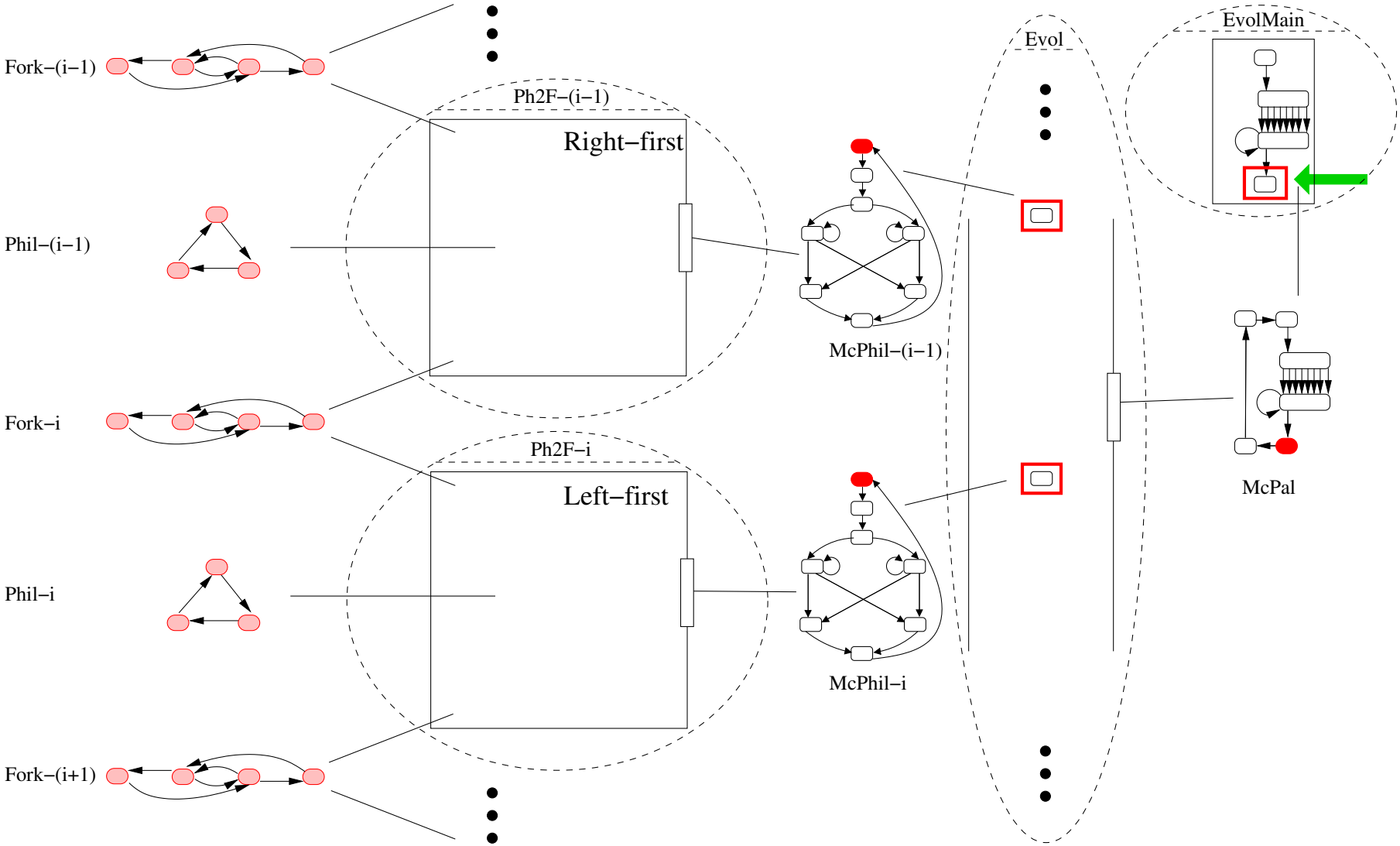
animation of as-is coordination

consistency rule

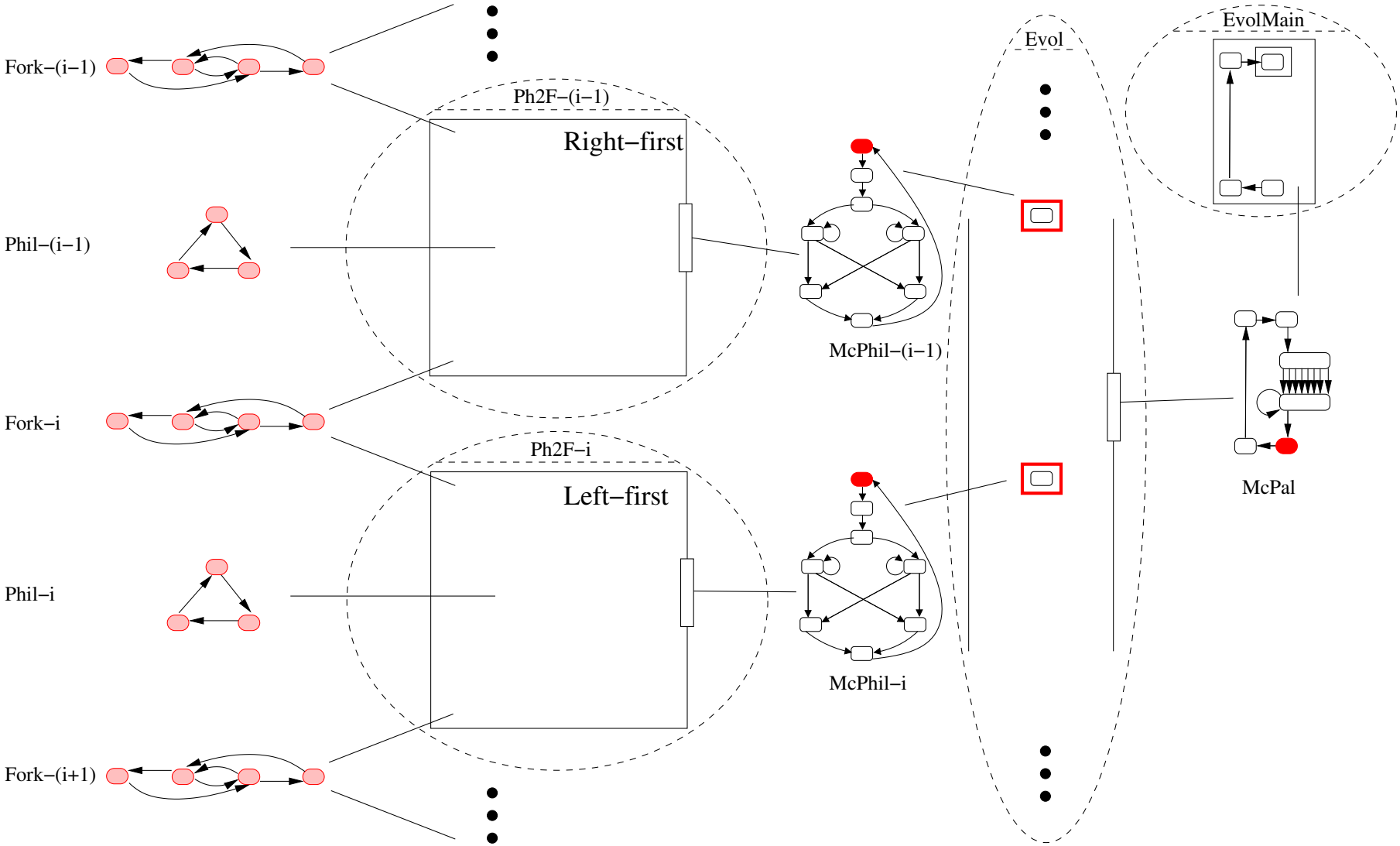
$*$ $McPal(Eater)$: $Migrating \xrightarrow{\text{done}} Hibernating$

is now applicable

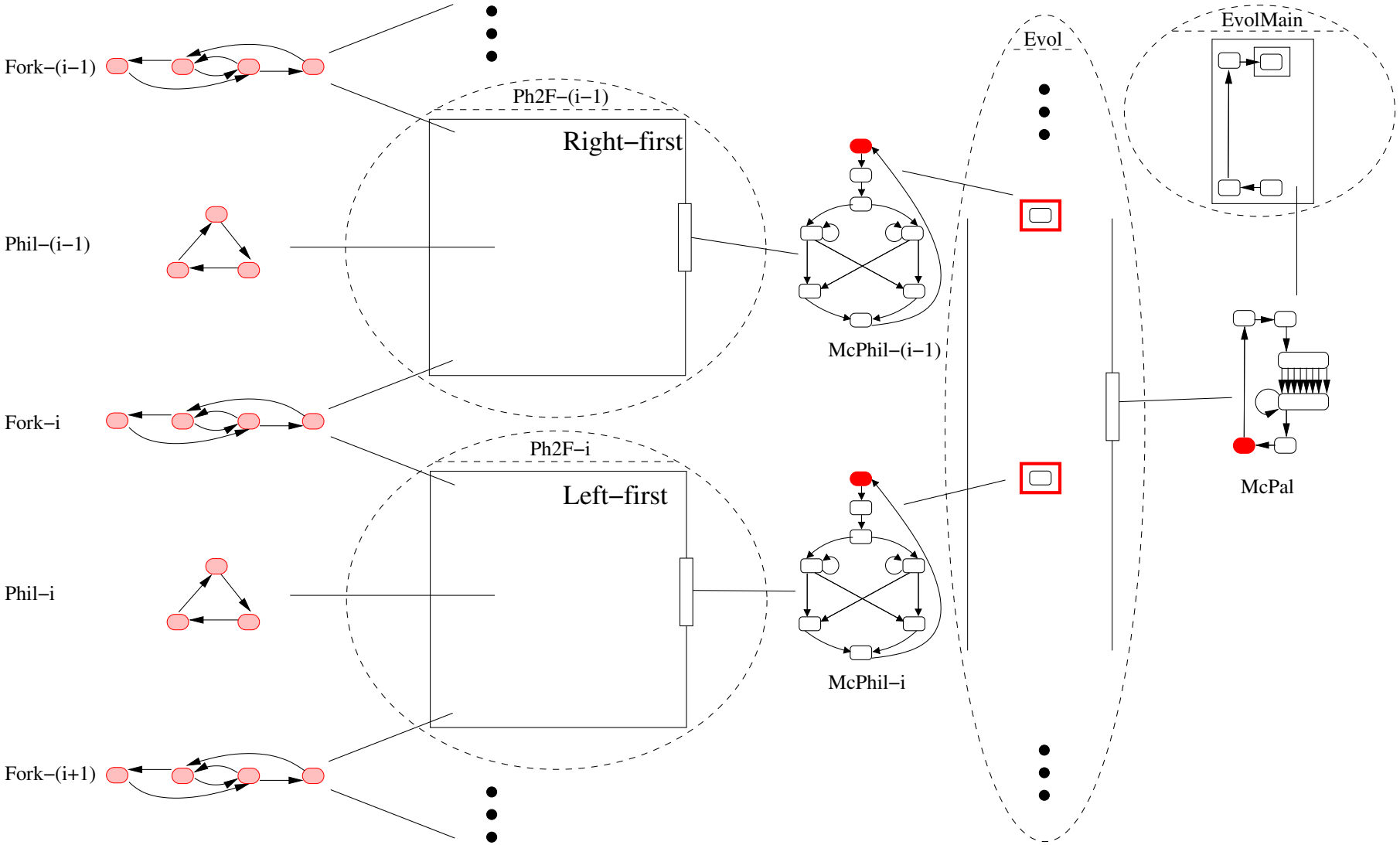
animation of migration coordination



animation of migration coordination



animation of migration coordination



animation of migration coordination

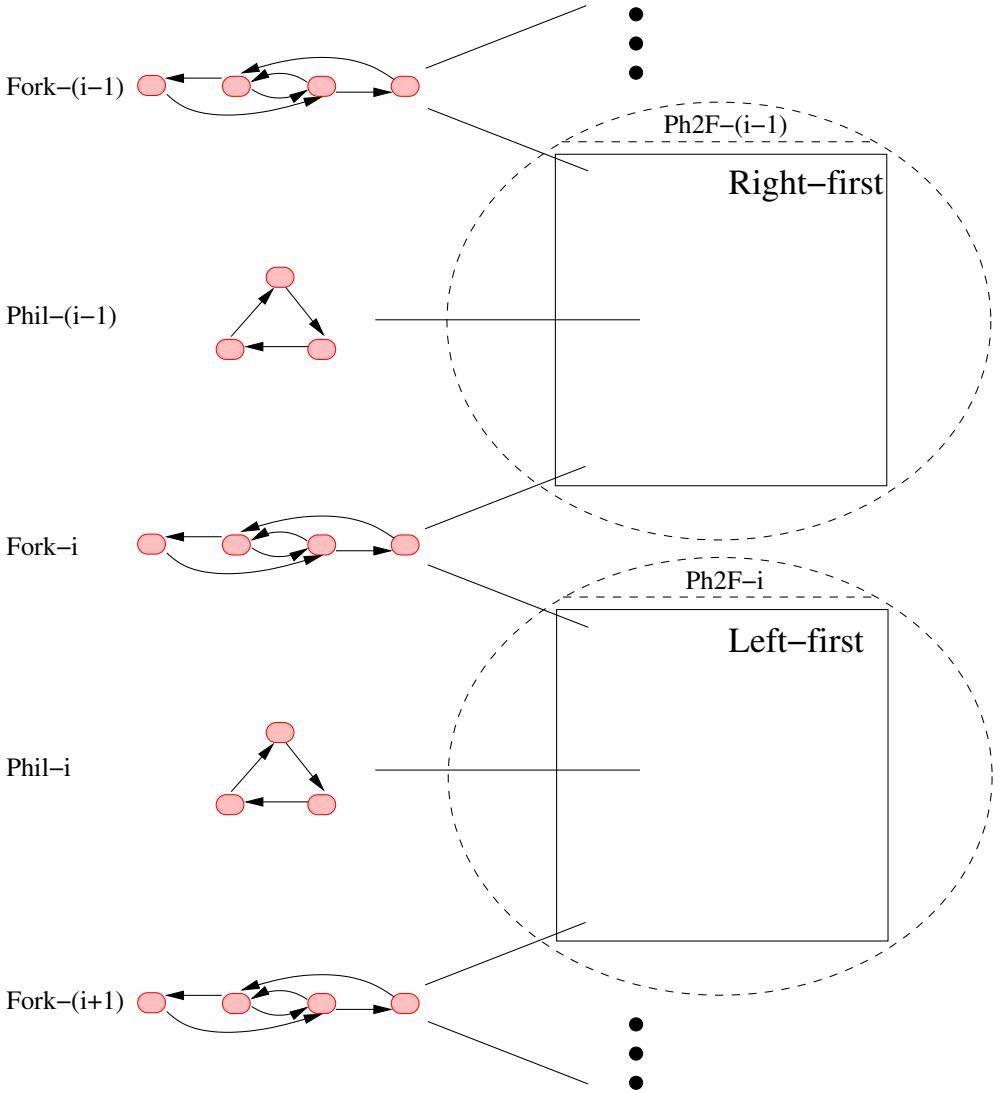
via consistency rule

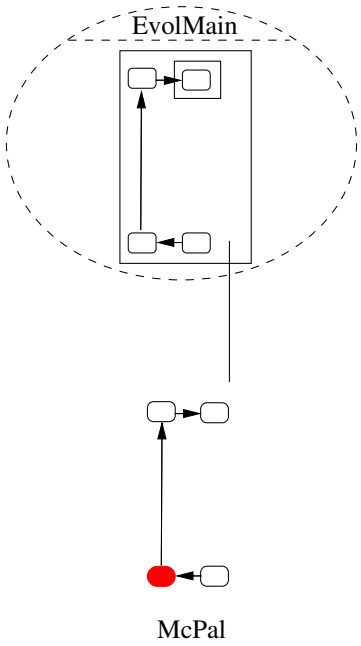
$$McPal : Content \xrightarrow{\text{cleanUp}} Observing *$$

$$McPal : [Crs := Crs_{toBe}]$$

McPal shrinks the model

animation of migration coordination





animation of migration coordination

. . . and so on and so forth . . .

concluding remarks

- distributed self-adaptation
- on-the-fly, without any quiescence
- creation / deletion of components
- computation of behaviour
- zipless swapping choreography – orchestration

future directions

- model checking with mCRL2 / Prism
- larger examples
- applications: Edafmis, network maintenance
- migration patterns