Event-B Course

3. File Transfer Protocol

Jean-Raymond Abrial

September-October-November 2011

Purpose of this Lecture

- To introduce another example: the file transfer protocol

- To present a number of additional mathematical conventions

- To slighly enlarge the usage of the Proof Obligation Rules

- Example studied in many places, in particular in the following book

- L. Lamport *Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers* Addison-Wesley 1999

Requirement Document (1)

- A file is to be transfered from a Sender to a Receiver

- On the Sender's side the file is called f

- On the Receiver's side the file is called g

- At the beginning of the protocol, g is supposed to be empty

- At the end of the protocol, g should be equal to f

Requirement Document (2)

The protocol ensures the copy of a file from one site to another one	FUN-1

The file is supposed to be made of a sequence of items	FUN-2

The file is sent piece by piece between the two sites	FUN-3

Modeling Approach

- Our approach at modeling is one of an external observer

- The observer “sees” the state space first from very far away

- He then approaches the future system and sees more details

- As he approaches he also sees more things happening

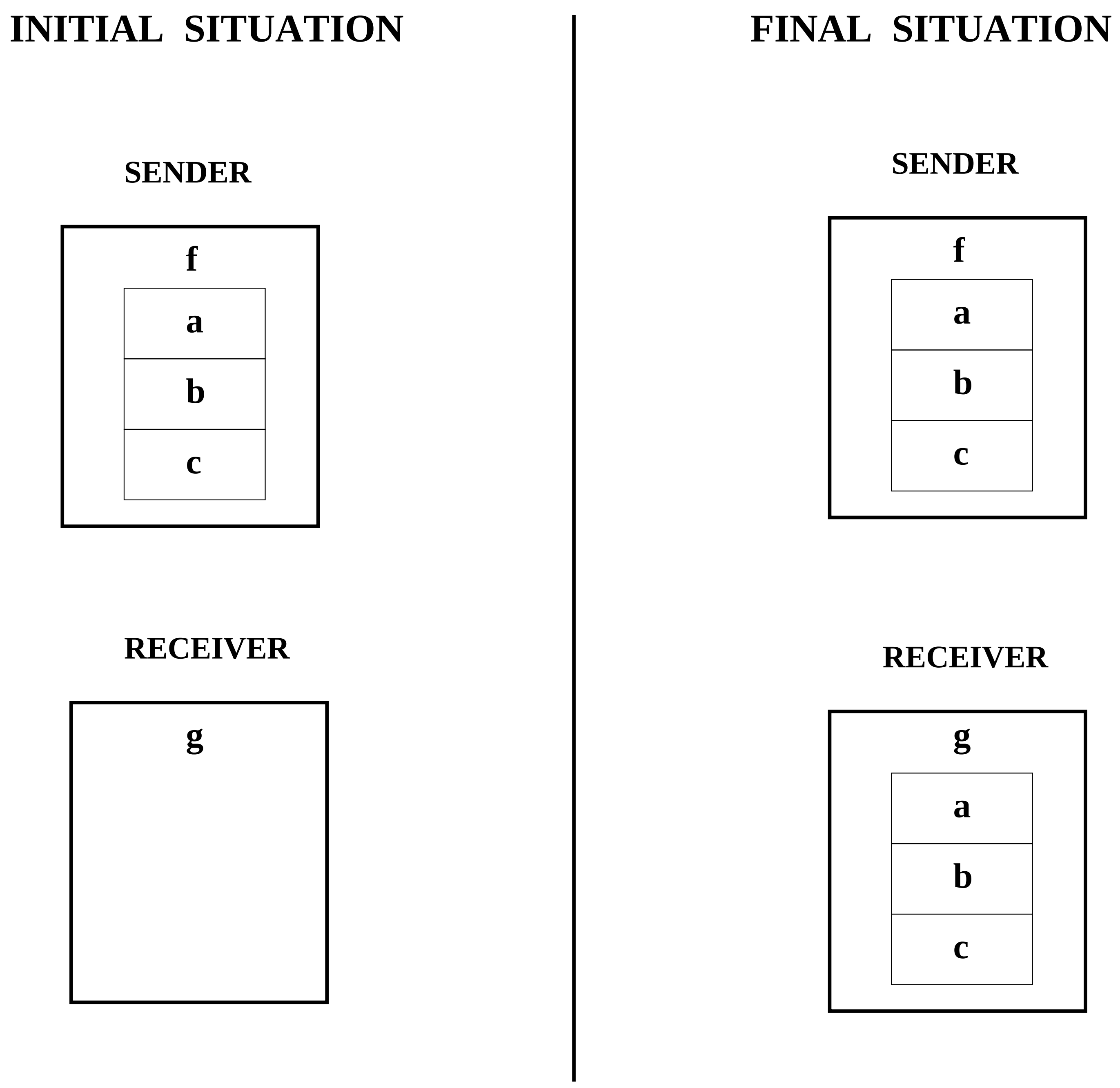
Refinement Strategy

- Initial model: The file is transmitted in one shot (FUN1 and FUN2)

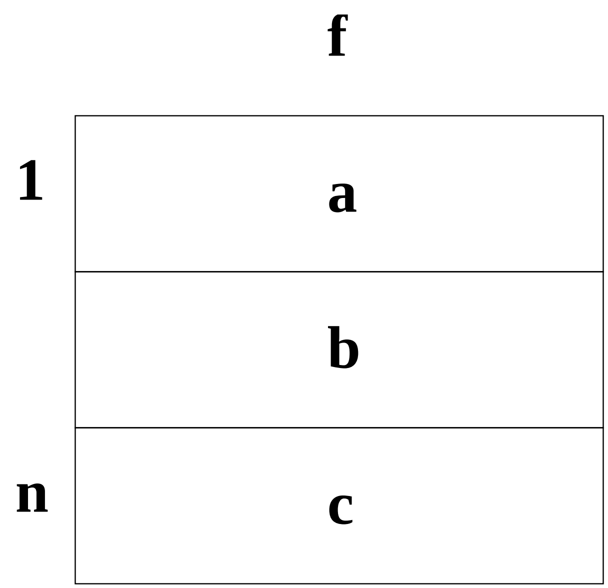
- First refinement: The file is transmitted gradually (FUN3)

- Second refinement: The two agents are separated

- Third refinement: Towards an implementation


INITIAL SITUATION
SENDER
f
a
b
c
RECEIVER
g
FINAL SITUATION
SENDER
f
a
b
c
RECEIVER
g
a
b
c

File transfer. The constant part of the state: n and f



The Initial State: Constant Part

sets: D

constants: n
f

axm0_1: $0 < n$

axm0_2: $f \in 1\,..\,n \rightarrow D$

- The carrier set D makes this development generic

The Initial State: Dynamic Part

variables: g
b

inv0_1: $g \in 1\,..\,n \nrightarrow D$

inv0_2: $b = \mathrm{FALSE} \;\Rightarrow\; g = \varnothing$

inv0_3: $b = \mathrm{TRUE} \;\Rightarrow\; g = f$

$b = \mathrm{TRUE}$ means protocol is finished

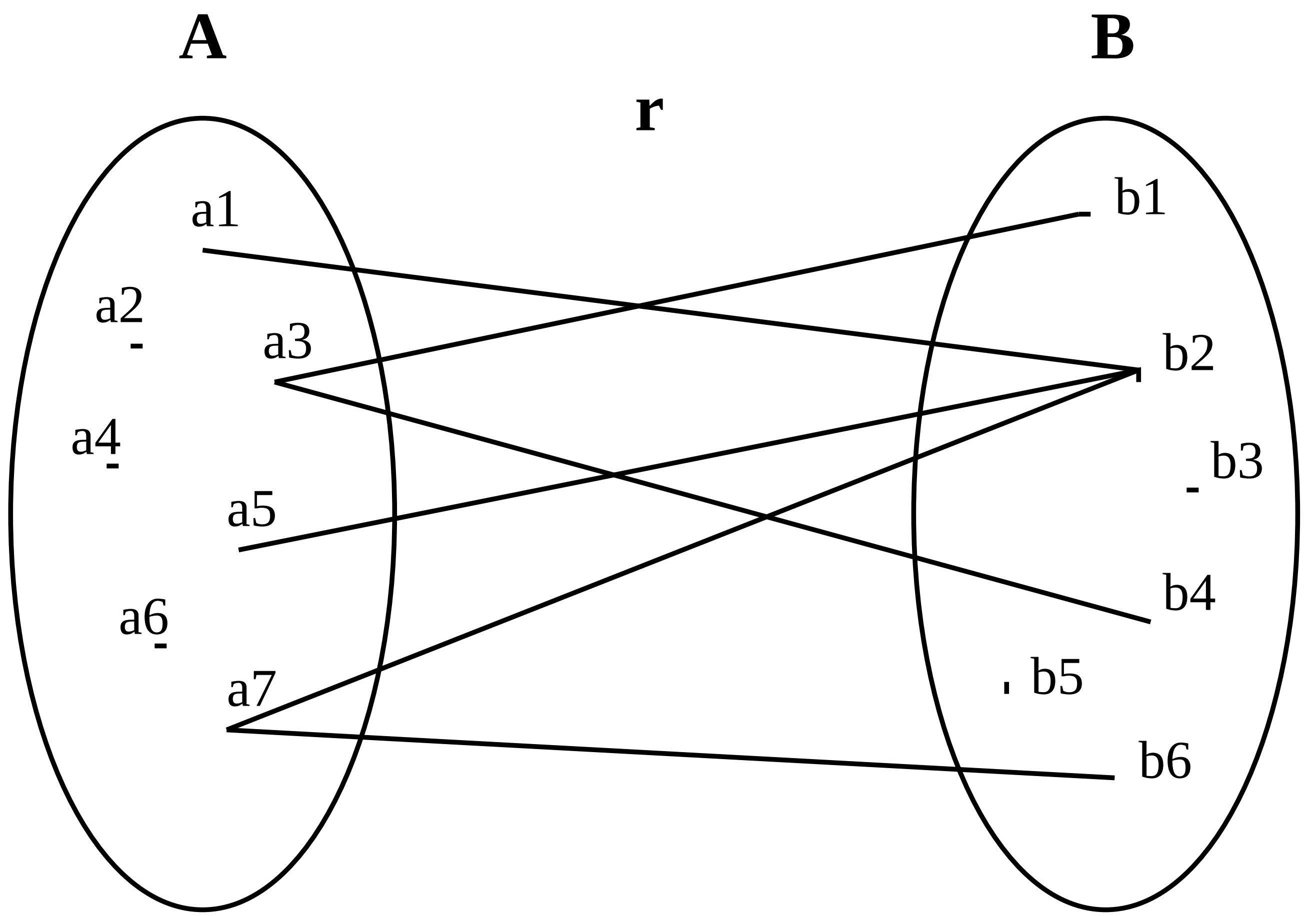
Reminder of Mathematical Conventions (1)

$x \in S$	set membership operator
$\mathbb{N}$	set of natural numbers: $\{0, 1, 2, 3, \ldots\}$
$a \mathbin{..} b$	interval from a to b: $\{a, a+1, \ldots, b\}$ (empty when $b < a$)
$a \mapsto b$	pair constructing operator
$S \times T$	Cartesian product operator
$S \subseteq T$	set inclusion operator
$\mathbb{P}(S)$	power set operator

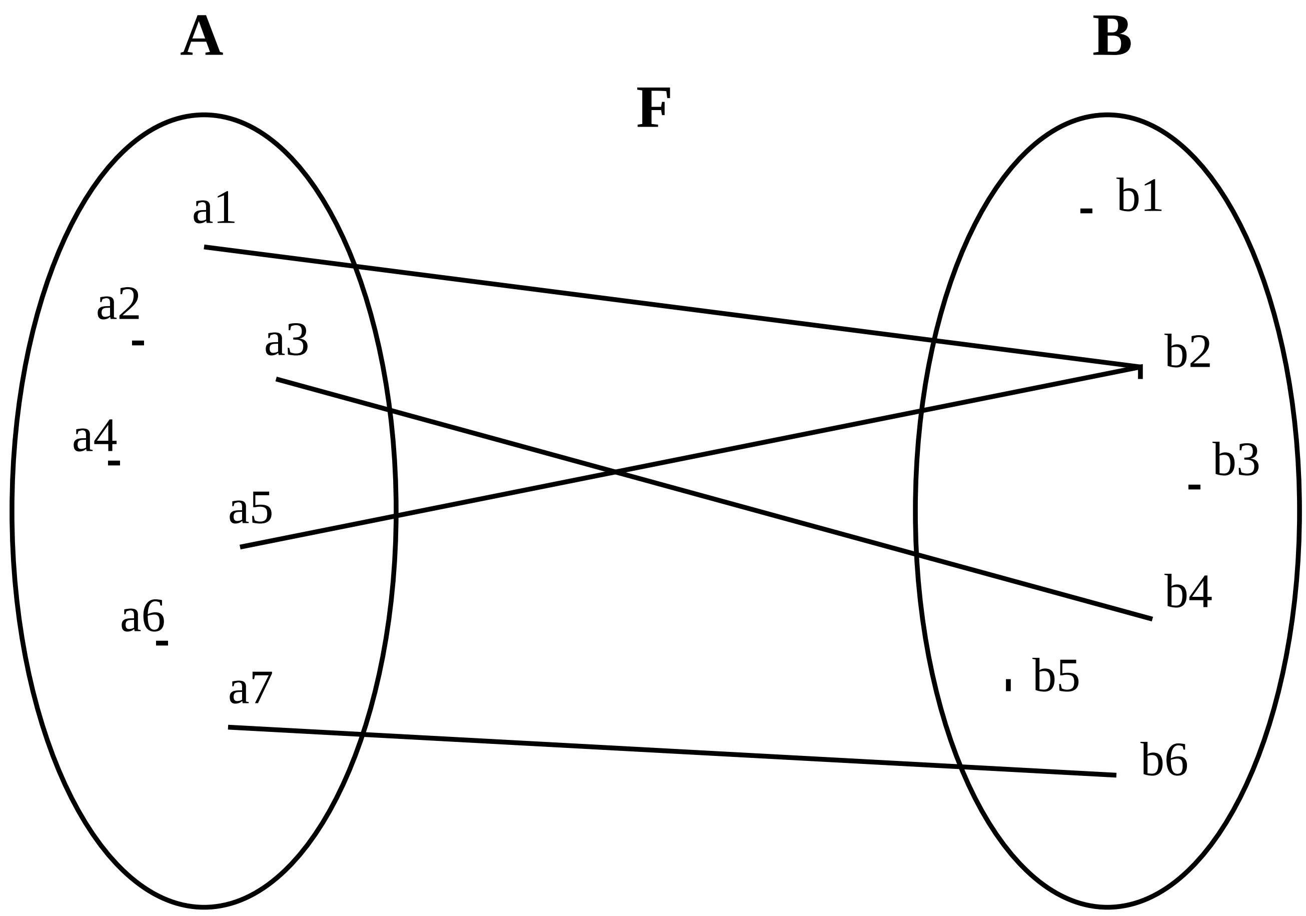
Reminder of Mathematical Conventions (2)

$S \leftrightarrow T$	set of binary relations from S to T
$S \rightarrow T$	set of total functions from S to T
$S \nrightarrow T$	set of partial functions from S to T
$\mathrm{dom}(r)$	domain of a relation r
$\mathrm{ran}(r)$	range of a relation r

A Binary Relation r from a Set A to a Set B

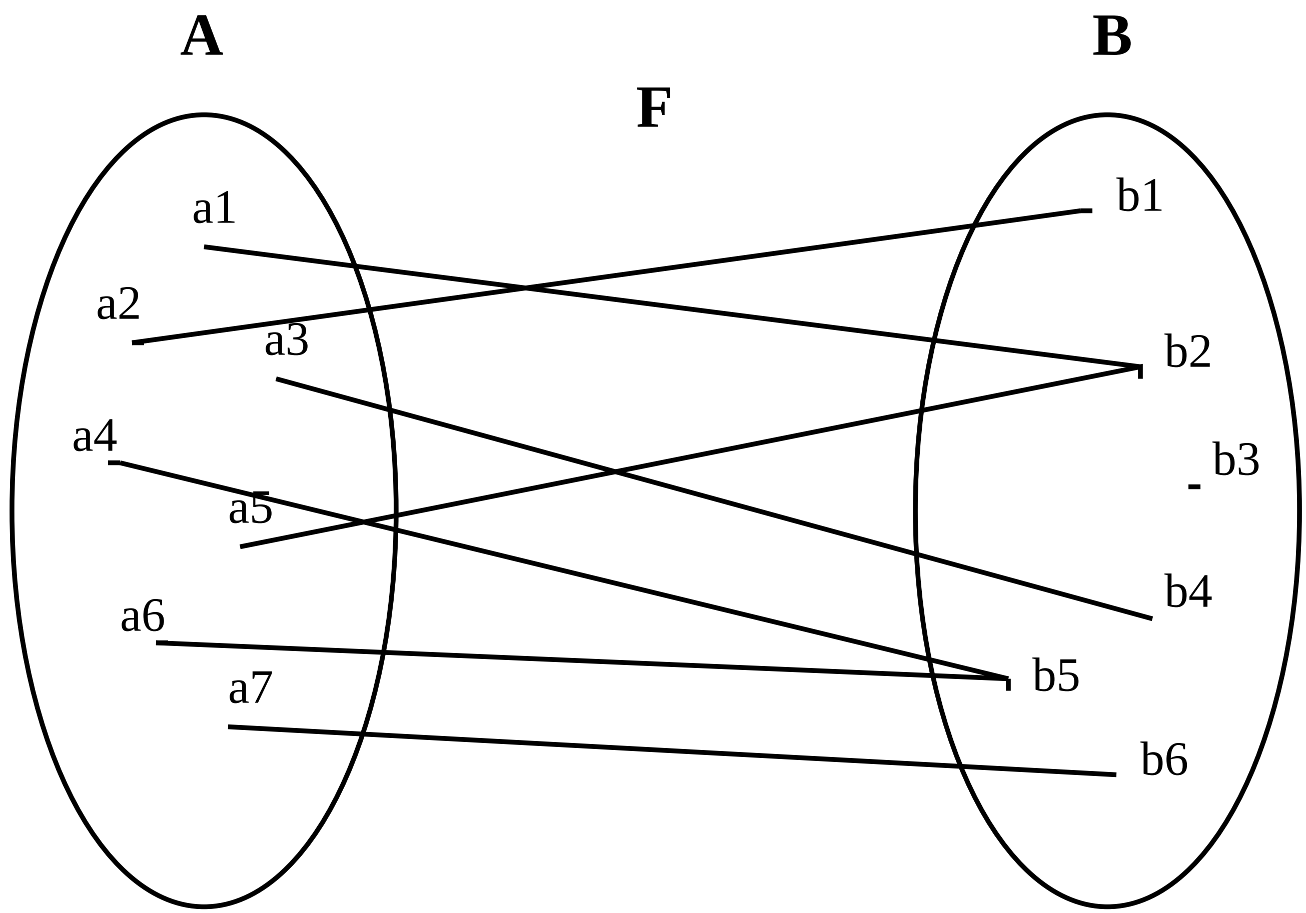


A Partial Function F from a Set A to a Set B



$$
\begin{aligned}
F &= \{a1 \mapsto b2,\ \ a3 \mapsto b4,\ \ a5 \mapsto b2,\ \ a7 \mapsto b6\} \\
\mathsf{dom}\,(F) &= \{a1,\ \ a3,\ \ a5,\ \ a7\} \\
\mathsf{ran}\,(F) &= \{b2,\ \ b4,\ \ b6\}
\end{aligned}
$$

A Total Function F from a Set A to a Set B



$$\mathsf{dom}\,(F) \;=\; A$$

File Transfer Protocol Events: init and final

init
$$g := \varnothing$$
$$b := \mathrm{FALSE}$$

final
when
$$b = \mathrm{FALSE}$$
then
$$g := f$$
$$b := \mathrm{TRUE}$$
end

sets: D

constants: n
f

axm0_1: $0 < n$

axm0_2: $f \in 1 \,..\, n \rightarrow D$

variables: g
b

inv0_1: $g \in 1 \,..\, n \leftrightarrow D$

inv0_2: $b = \mathrm{FALSE} \;\Rightarrow\; g = \varnothing$

inv0_3: $b = \mathrm{TRUE} \;\Rightarrow\; g = f$

init
$g := \varnothing$
$b := \mathrm{FALSE}$

final
when
$b = \mathrm{FALSE}$
then
$g := f$
$b := \mathrm{TRUE}$
end

What is to be Proved

- Event init establishes invariants **inv0_1** to **inv0_3** (Rule INV)

- Event final preserves invariants **inv0_1** to **inv0_3** (Rule INV)

inv0_1: $g \in 1 \,..\, n \nrightarrow D$

inv0_2: $b = \mathrm{FALSE} \;\Rightarrow\; g = \varnothing$

inv0_3: $b = \mathrm{TRUE} \;\Rightarrow\; g = f$

init
  $g := \varnothing$
  $b := \mathrm{FALSE}$

final
  when
    $b = \mathrm{FALSE}$
  then
    $g := f$
    $b := \mathrm{TRUE}$
  end

Invariant Establishment Proof Rule

- For the init event in the initial model

Properties $\vdash$ Modified Invariant	INV

Proof of Event init (1)

- Applying Rule INV to invariant **inv0_1**

init
$g := \varnothing$
$b := \mathrm{FALSE}$

inv0_1: $g \in 1 \,..\, n \leftrightarrow D$ (partial function arrow)

axm0_1
axm0_2
$\vdash$
modified **inv0_1**

$$0 < n$$
$$f \in 1 \,..\, n \to D$$
$$\vdash$$
$$\varnothing \in 1 \,..\, n \nrightarrow D$$

inv0_1 / INV

Proof of Event init (2)

- Applying Rule INV to invariant **inv0_2**

init
$g := \varnothing$
$b := \mathrm{FALSE}$

inv0_2: $b = \mathrm{FALSE} \;\Rightarrow\; g = \varnothing$

axm0_1
axm0_2
$\vdash$
modified **inv0_2**

$$
\begin{array}{l}
0 < n \\
f \in 1\,..\,n \rightarrow D \\
\vdash \\
\mathrm{FALSE} = \mathrm{FALSE} \;\Rightarrow\; \varnothing = \varnothing
\end{array}
$$

inv0_2 / INV

Proof of Event init (3)

- Applying Rule INV to invariant **inv0_3**

init
$g := \varnothing$
$b := \mathrm{FALSE}$

inv0_3: $b = \mathrm{TRUE} \;\Rightarrow\; g = f$

axm0_1 **axm0_2** $\vdash$ modified **inv0_3**	$0 < n$ $f \in 1 \,..\, n \to D$ $\vdash$ $\mathrm{FALSE} = \mathrm{TRUE} \;\Rightarrow\; \varnothing = f$	**inv0_3** / INV

Invariant Preservation Proof Rule

- For other events in the initial model

Properties Invariants Guards of the event $\vdash$ Modified Invariant	INV

Proof of Event final (1)

- Applying Rule INV

final
  when
    $b = \mathrm{FALSE}$
  then
    $g := f$
    $b := \mathrm{TRUE}$
  end

axm0_1 **axm0_2** **inv0_1** $\ldots$ **grd** $\vdash$ modified **inv0_1**	$0 < n$ $f \in 1\,..\,n \rightarrow D$ $g \in 1\,..\,n \nrightarrow D$ $\ldots$ $b = \mathrm{FALSE}$ $\vdash$ $f \in 1\,..\,n \nrightarrow D$	final / **inv0_1** / INV

Proof of Event final (2)

- Applying Rule INV

final
  when
    $b = \mathrm{FALSE}$
  then
    $g := f$
    $b := \mathrm{TRUE}$
  end

axm0_1	$0 < n$
axm0_2	$f \in 1 \,..\, n \rightarrow D$
inv0_2	$b = \mathrm{FALSE} \Rightarrow g = \varnothing$
$\ldots$	$\ldots$
grd	$b = \mathrm{FALSE}$
$\vdash$	$\vdash$
modified **inv0_2**	$\mathrm{TRUE} = \mathrm{FALSE} \Rightarrow f = \varnothing$

final / **inv0_2** / INV

Proof of Event final (3)

- Applying Rule INV

final
  when
    $b = \mathrm{FALSE}$
  then
    $g := f$
    $b := \mathrm{TRUE}$
  end

axm0_1	$0 < n$	
axm0_2	$f \in 1\,..\,n \rightarrow D$	
inv0_3	$b = \mathrm{TRUE} \Rightarrow g = f$	
$\ldots$	$\ldots$	final / **inv0_3** / INV
grd	$b = \mathrm{FALSE}$	
$\vdash$	$\vdash$	
modified **inv0_3**	$\mathrm{TRUE} = \mathrm{TRUE} \Rightarrow f = f$	

Refinement Strategy

- Initial model: The file is transmitted in one shot (FUN1 and FUN2)

- First refinement: The file is transmitted gradually (FUN3)

- Second refinement: The two agents are separated

- Third refinement: Towards an implementation

- The observer comes closer to the future system

- So far he was just seeing the beginning and the end

- Now the observer will see some intermediate moves

- He sees the file being gradually transfered from Sender to Receiver

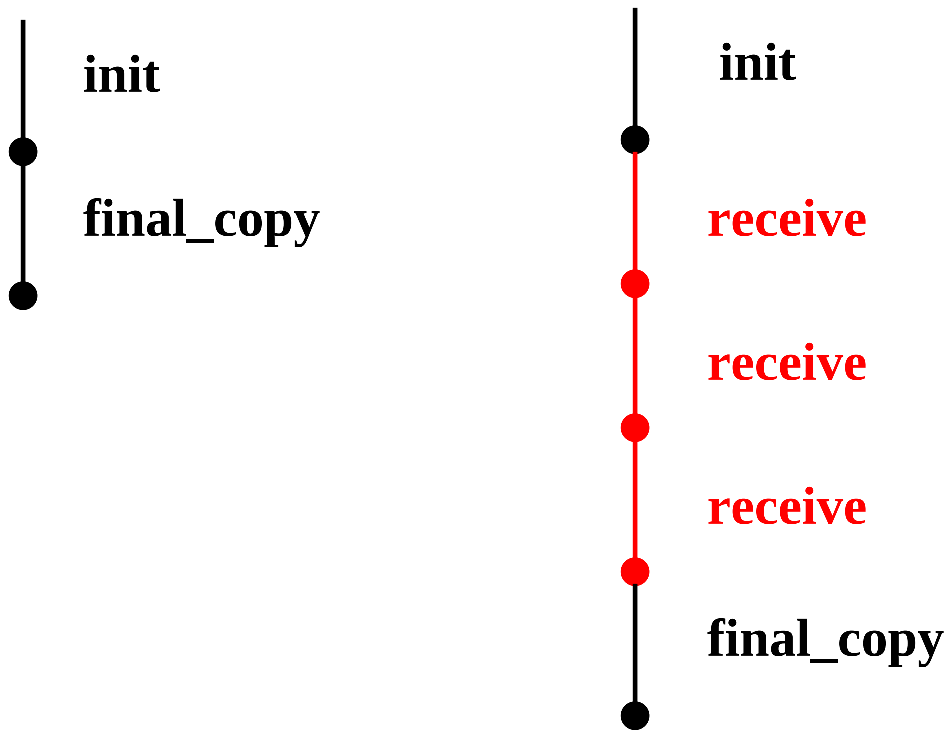
- But he still has a partial view

What the Observer has Seen so far

init

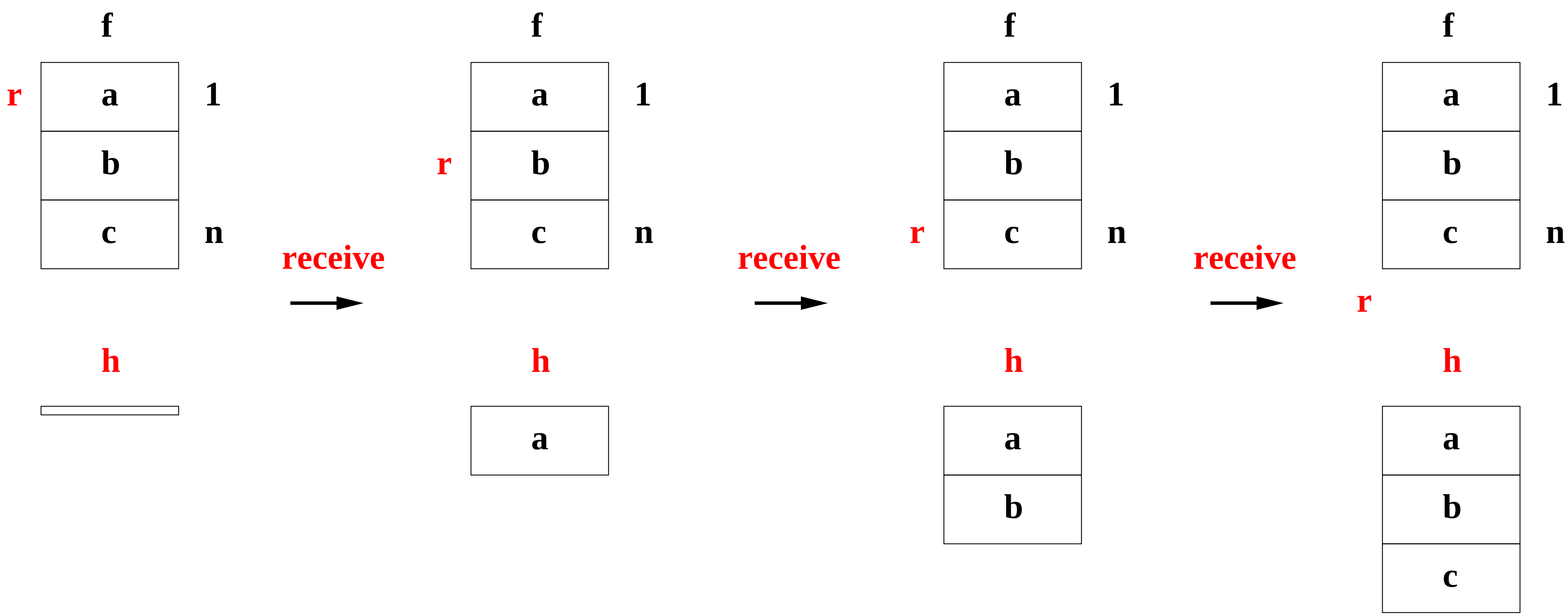
final_copy

What the Observer will now See



A new event is introduced: receive

File transfer. Event receive



- The new variable r lies within the interval $1\,..\,n+1$

- The new variable h is equal to f restricted to its $r-1$ first values

First Refinement of the Protocol Model: the State

- Introducing additional variables h and r

- Variable g disappears

variables: b
h
r

inv1_1: $r \in 1 \,..\, n+1$

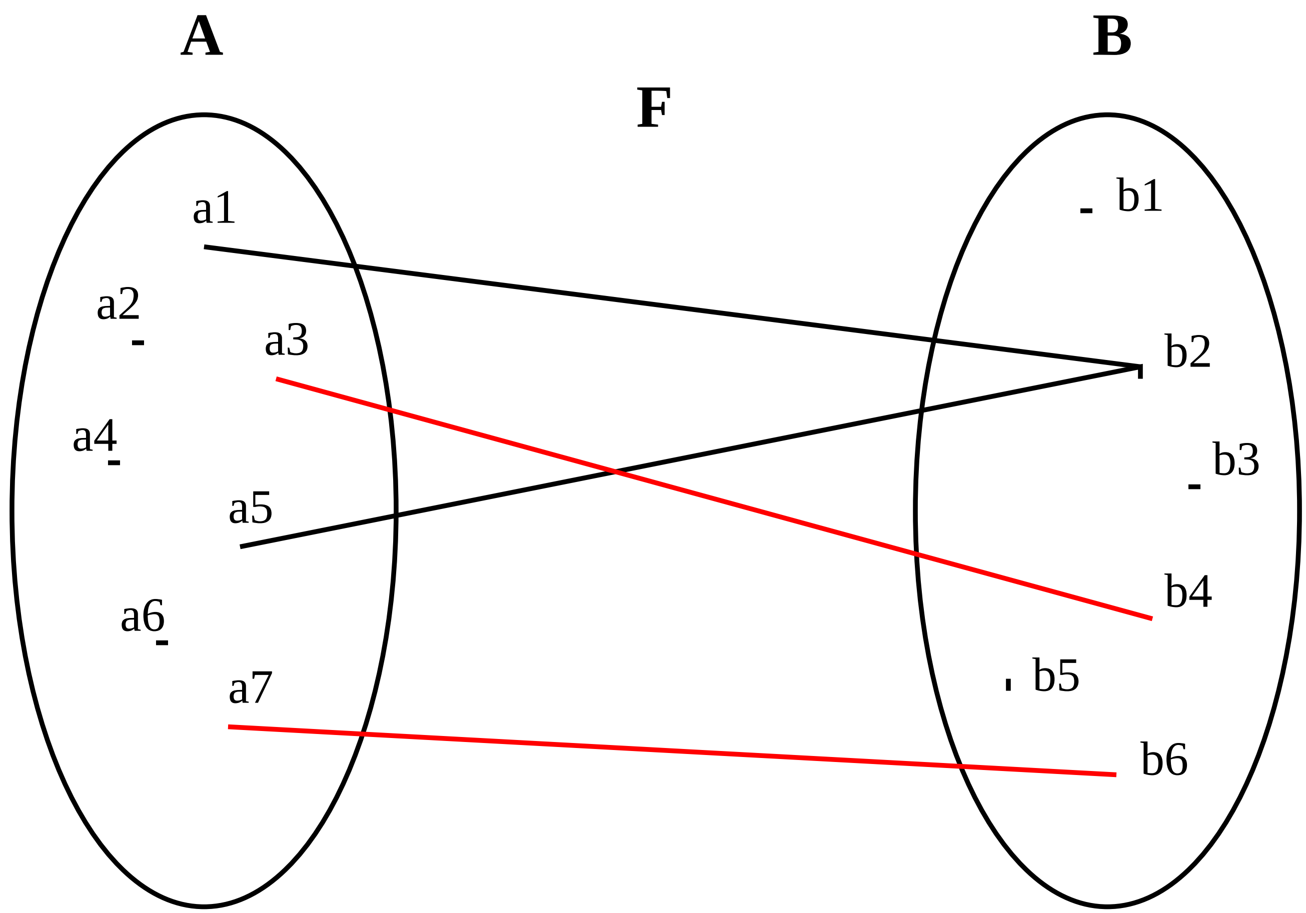
inv1_2: $h = (1 \,..\, r-1) \lhd f$

inv1_3: $b = \mathrm{TRUE} \Rightarrow r = n+1$

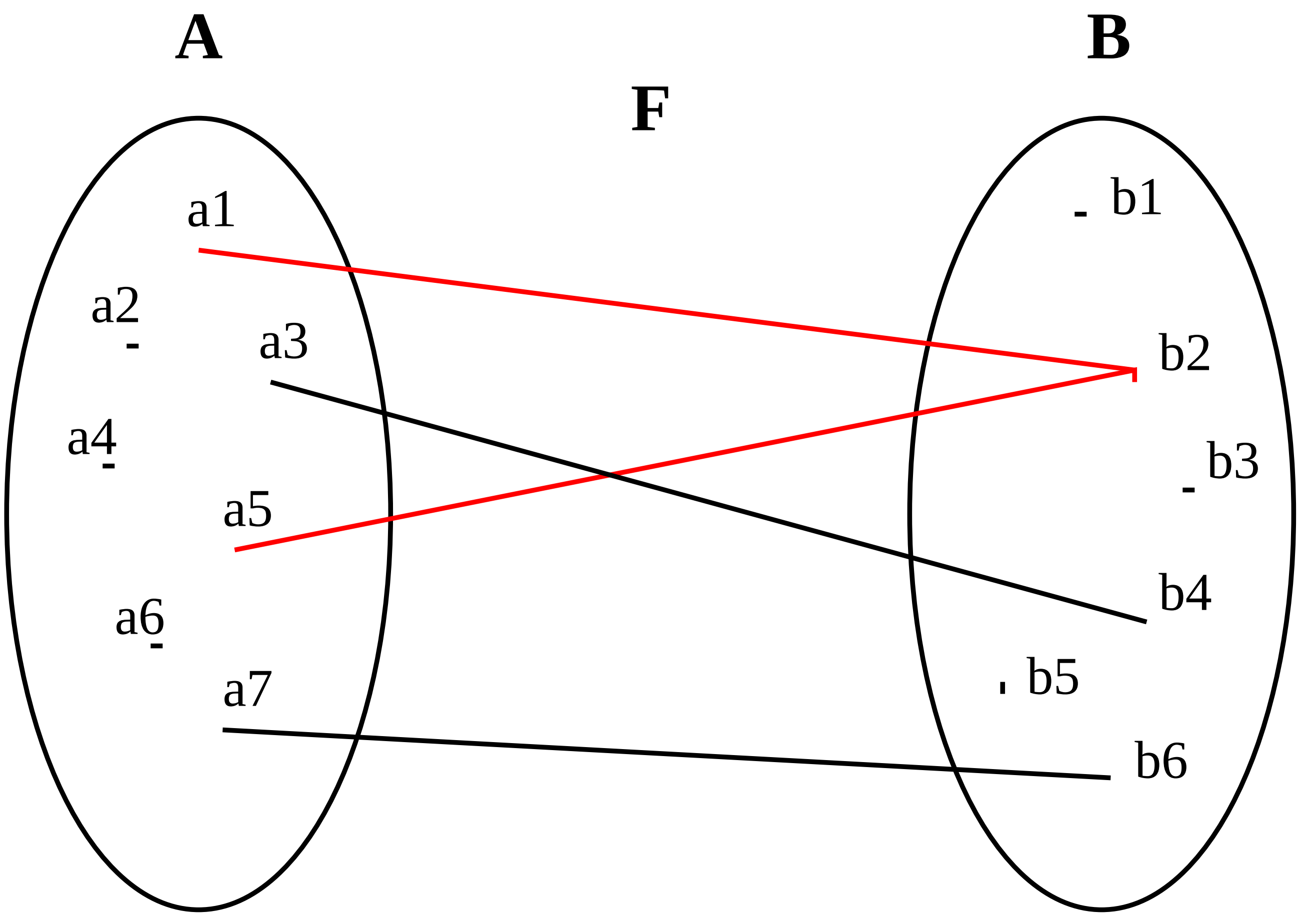
- h is defined to be the domain restriction of f to $1 \,..\, r-1$

More Mathematical Conventions: Restrictions

$s \lhd r$	domain restriction operator
$s \mathbin{\lhd\!\!\!-} r$	domain subtraction operator
$r \rhd t$	range restriction operator
$r \mathbin{\rhd\!\!\!-} t$	range subtraction operator

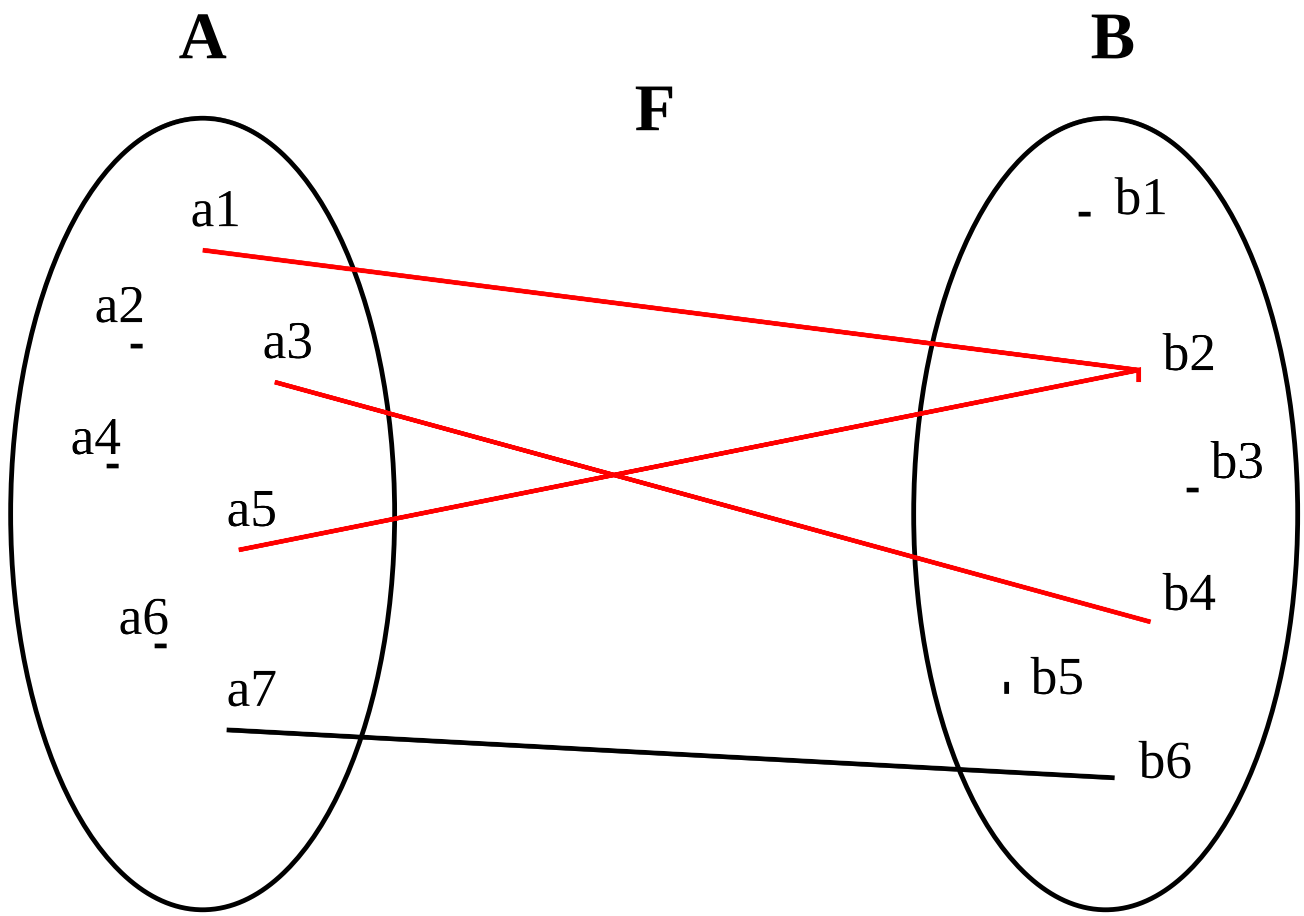


$$\{a3,\ a7\} \lhd F$$

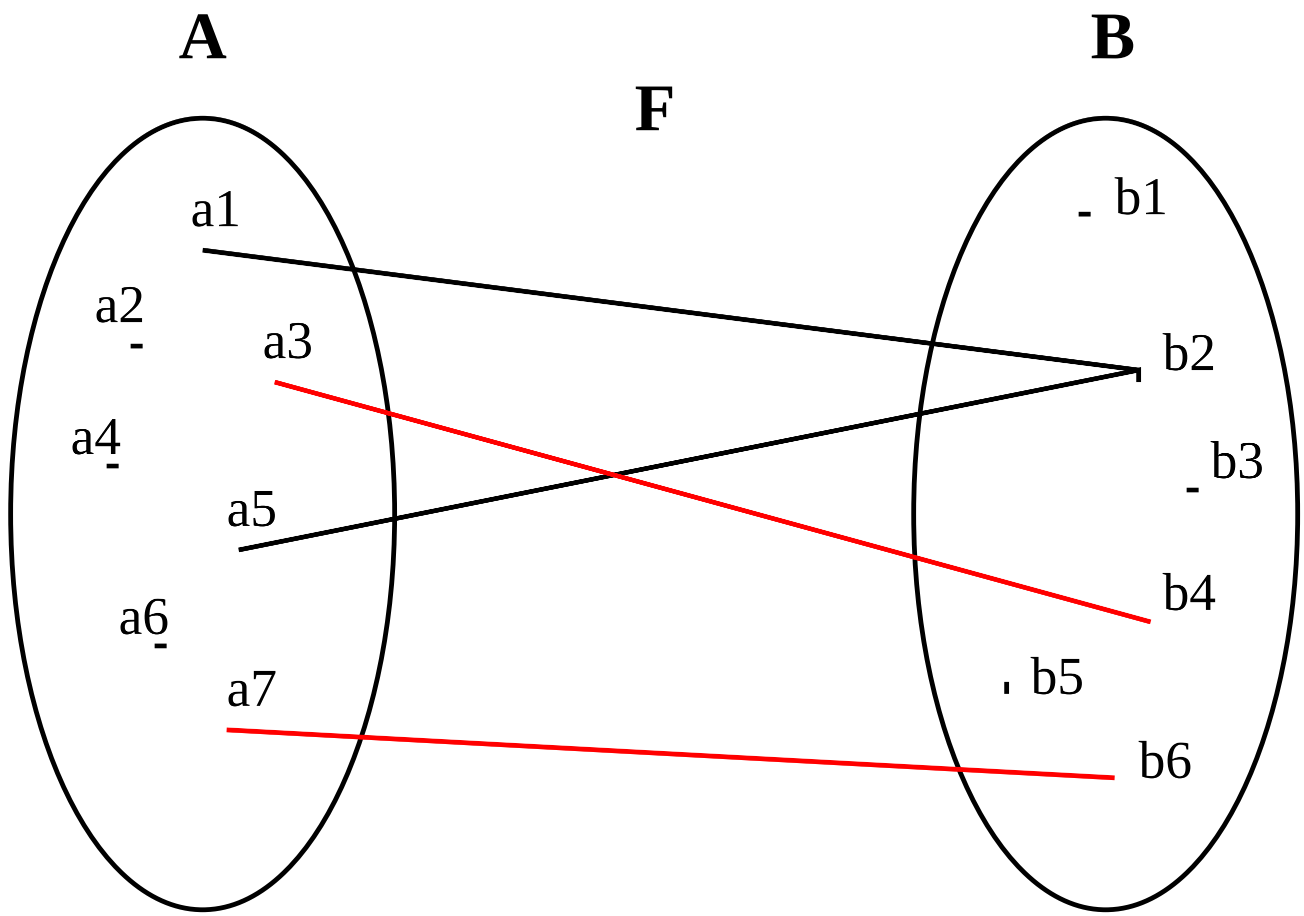


$$\{a3,\ a7\} \lhd F$$

The Range Restriction Operator



$$F \rhd \{b2, b4\}$$



$F \mathbin{\rhd\!\!\!-} \{b2\}$

init
$$h := \varnothing$$
$$r := 1$$
$$b := \mathrm{FALSE}$$

receive
when
$$r \leq n$$
then
$$h := h \cup \{r \mapsto f(r)\}$$
$$r := r + 1$$
end

final
when
$$r = n + 1$$
$$b = \mathrm{FALSE}$$
then
$$b := \mathrm{TRUE}$$
end

What is to be Proved

- Event init refines its abstraction

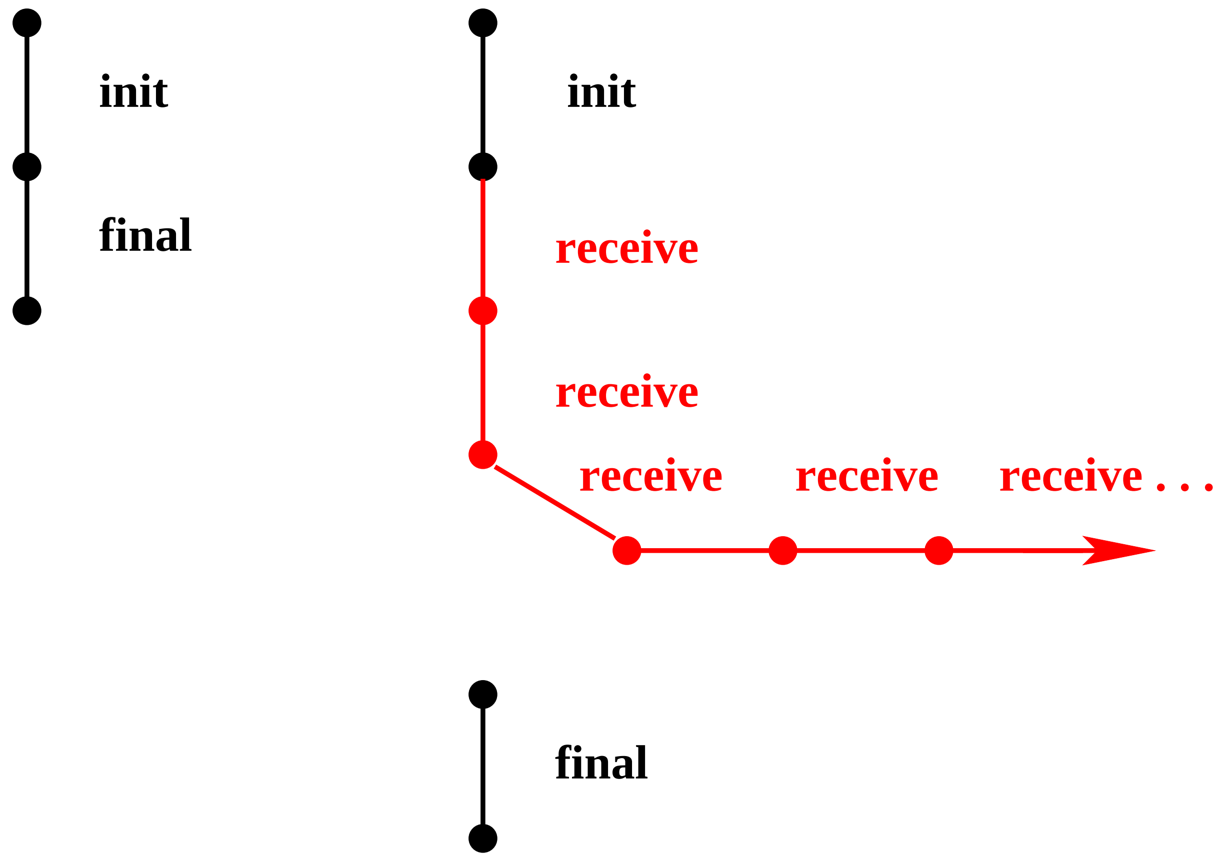
- Event final refines its abstraction

- Event receive refines skip

- Event receive does not diverge

- Relative deadlock freeness

First Pathology: Divergence



- No divergence of new event receive (rules NAT and VAR)

variant1: $n + 1 - r$

- This variant must be decreased by the new event:

receive
  when
    $r \leq n$
  then
    $h := h \cup \{r \mapsto f(r)\}$
    $r := r + 1$
  end

Refinement Rules: Non-divergence of New Events

- For new events only

Properties of the constants Abstract invariants Concrete invariants Concrete guards of a new event $\vdash$ Variant $\in \mathbb{N}$	NAT

- Applying rule NAT

...	...
inv1_1	$r \in 1 \,..\, n+1$
...	...
$\vdash$	$\vdash$
variant belongs to $\mathbb{N}$	$n+1-r \;\in\; \mathbb{N}$

- For new events only

Properties of the constants Abstract invariants Concrete invariants Concrete guards of a new event $\vdash$ Modified variant $<$ Variant	VAR

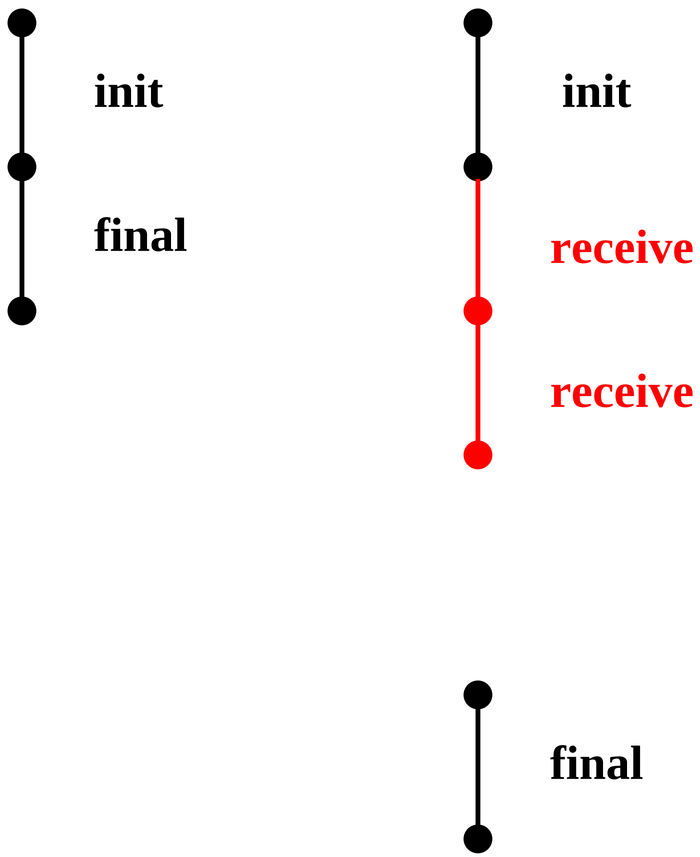
- Applying rule VAR

receive
  when
    $r \leq n$
  then
    $h := h \cup \{r \mapsto f(r)\}$
    $r := r + 1$
  end

$\ldots$
$\vdash$
variant is decreased

$\ldots$
$\vdash$
$n + 1 - (r + 1) < n + 1 - r$

Second Pathology: Early Deadlock



Refinement Rules: Relative Deadlock Freeness

- Global proof rule

Properties of the constants Abstract invariants Concrete invariants Disjunction of abstract guards $\vdash$ Disjunction of concrete guards	DLF

The Abstract and Concrete Events

- Abstract Events

final
  when
    $b = \mathrm{FALSE}$
  then
    $g := f$
    $b := \mathrm{TRUE}$
  end

- Concrete Events

receive
  when
    $r \leq n$
  then
    $h := h \cup \{r \mapsto f(r)\}$
    $r := r + 1$
  end

final
  when
    $b = \mathrm{FALSE}$
    $r = n + 1$
  then
    $b := \mathrm{TRUE}$
  end

More to be Proved: No Early Deadlock

- Applying rule DLF

$\dots$ **inv1_1** disj. of abs. guards $\vdash$ disj. of conc. guards	$\dots$ $r \in 1 \,..\, n+1$ $b = \mathrm{FALSE}$ $\vdash$ $r \leq n \;\; \lor \;\; (b = \mathrm{FALSE} \;\land\; r = n+1)$

variables: b
h
r

inv1_1: $r \in 1 \,..\, n+1$

inv1_2: $h = (1 \,..\, r-1) \lhd f$

inv1_3: $b = \mathrm{TRUE} \Rightarrow r = n+1$

variant1: $n+1-r$

init
$b := \mathrm{FALSE}$
$h := \varnothing$
$r := 1$

receive
when
$r \leq n$
then
$h := h \cup \{r \mapsto f(r)\}$
$r := r+1$
end

final
when
$b = \mathrm{FALSE}$
$r = n+1$
then
$b := \mathrm{TRUE}$
end

- This model is not satisfactory: event receive accesses file f

Development Approach

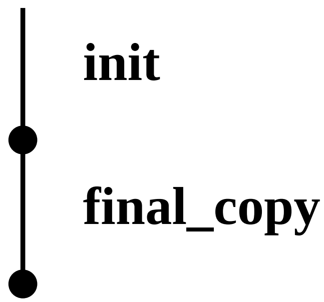
- Initial model: The file is transmitted in one shot (FUN1 and FUN2)

- First refinement: The file is transmitted gradually (FUN3)

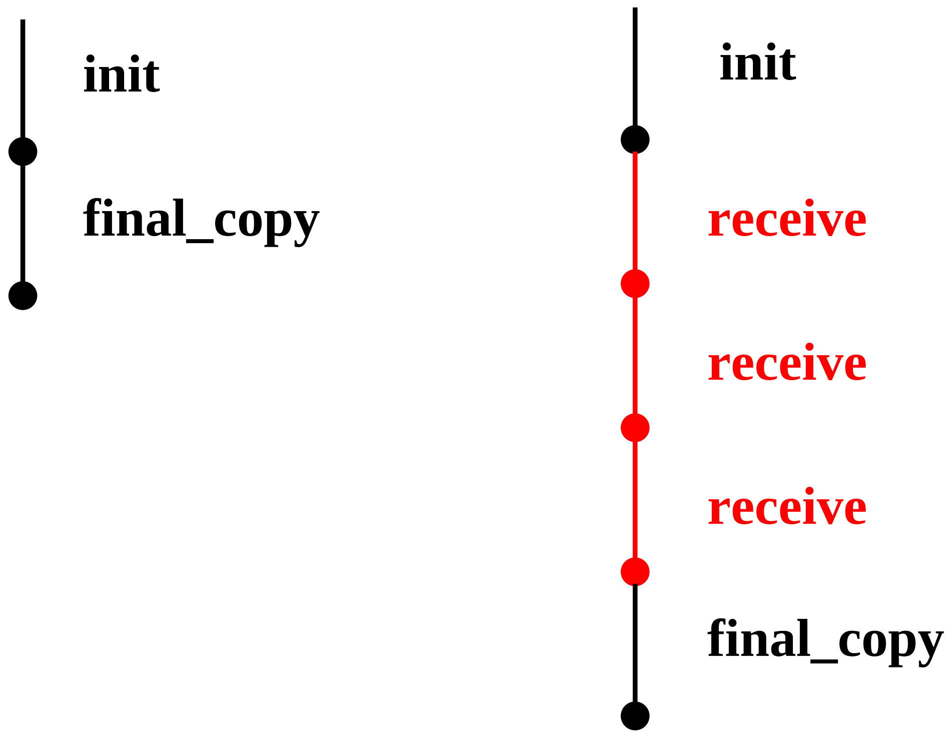
- Second refinement: The two agents are separated

- Third refinement: Towards an implementation

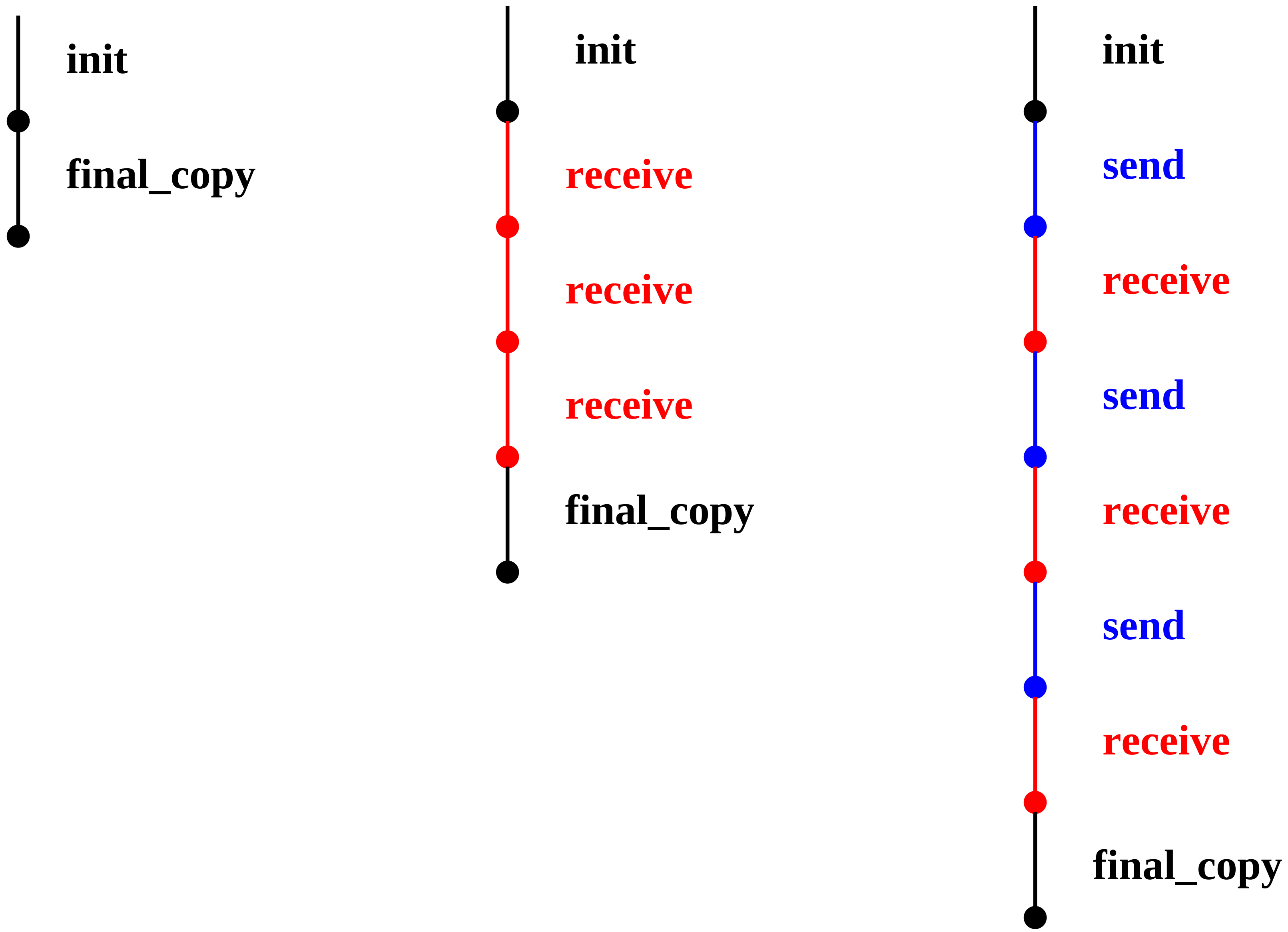
What the Observer could see of the Initial Model



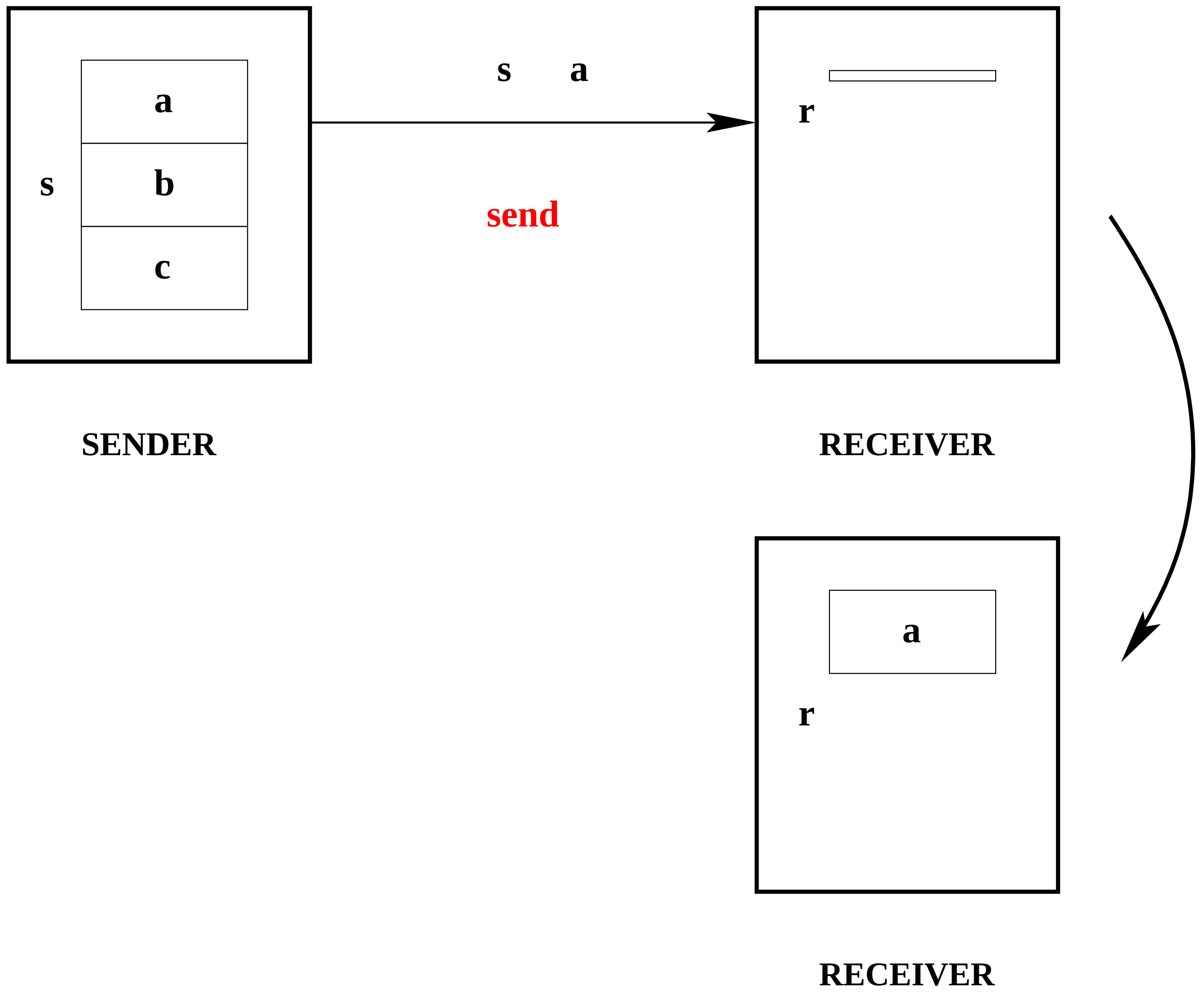
What the Observer could see of first Refinement



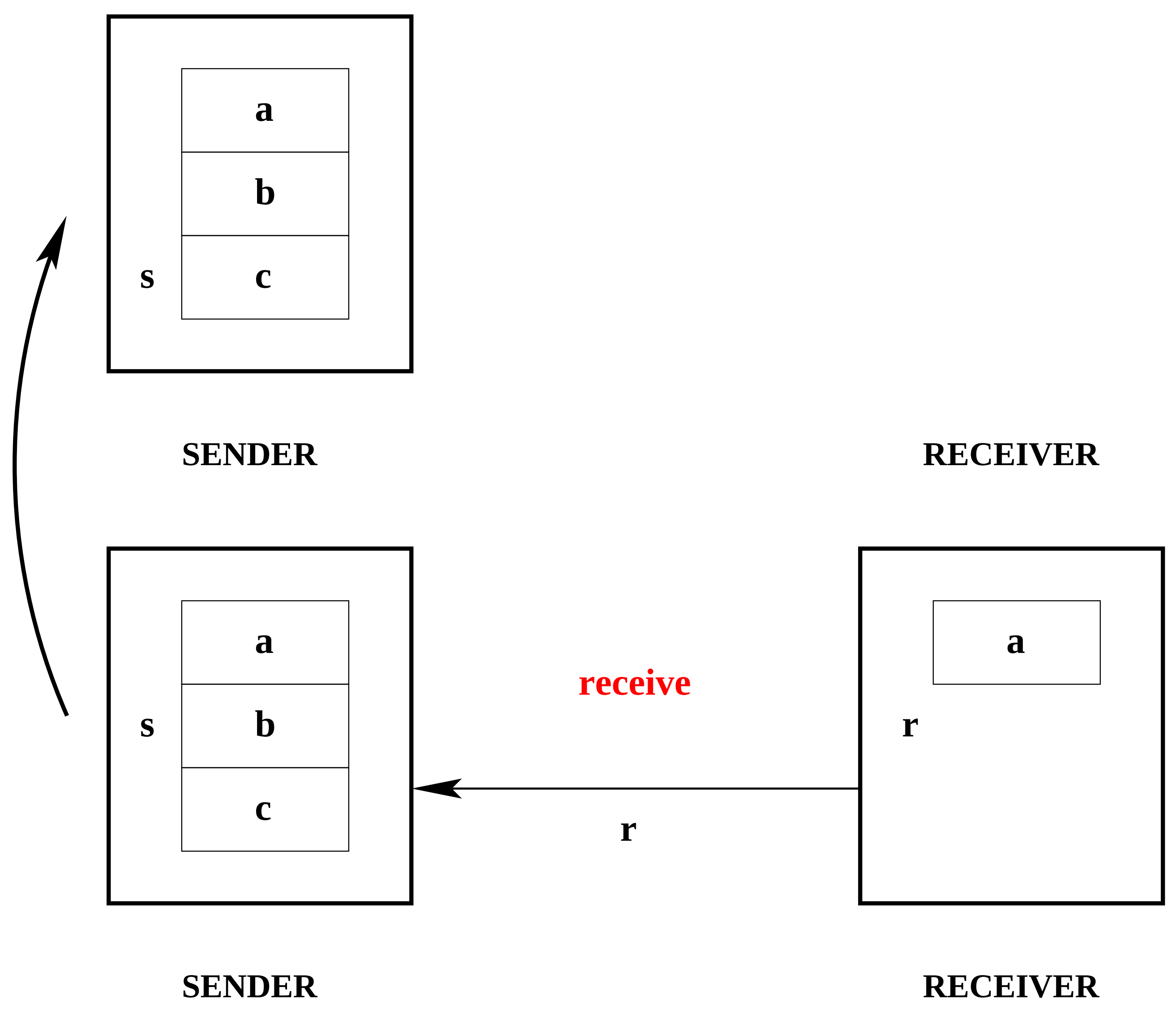
What the Observer will now See



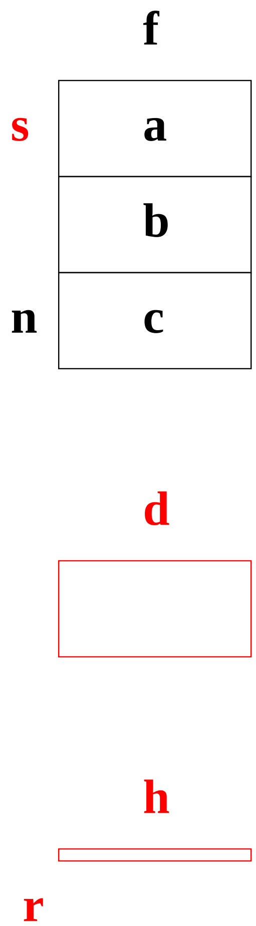
A More Accurate Version (1)

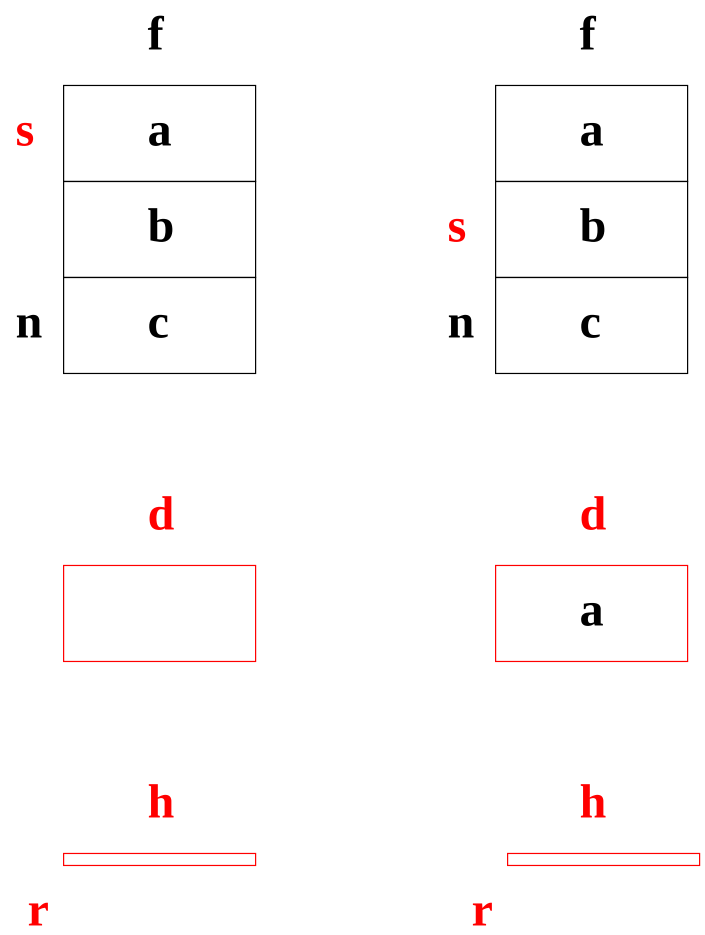


A More Accurate Version (2)

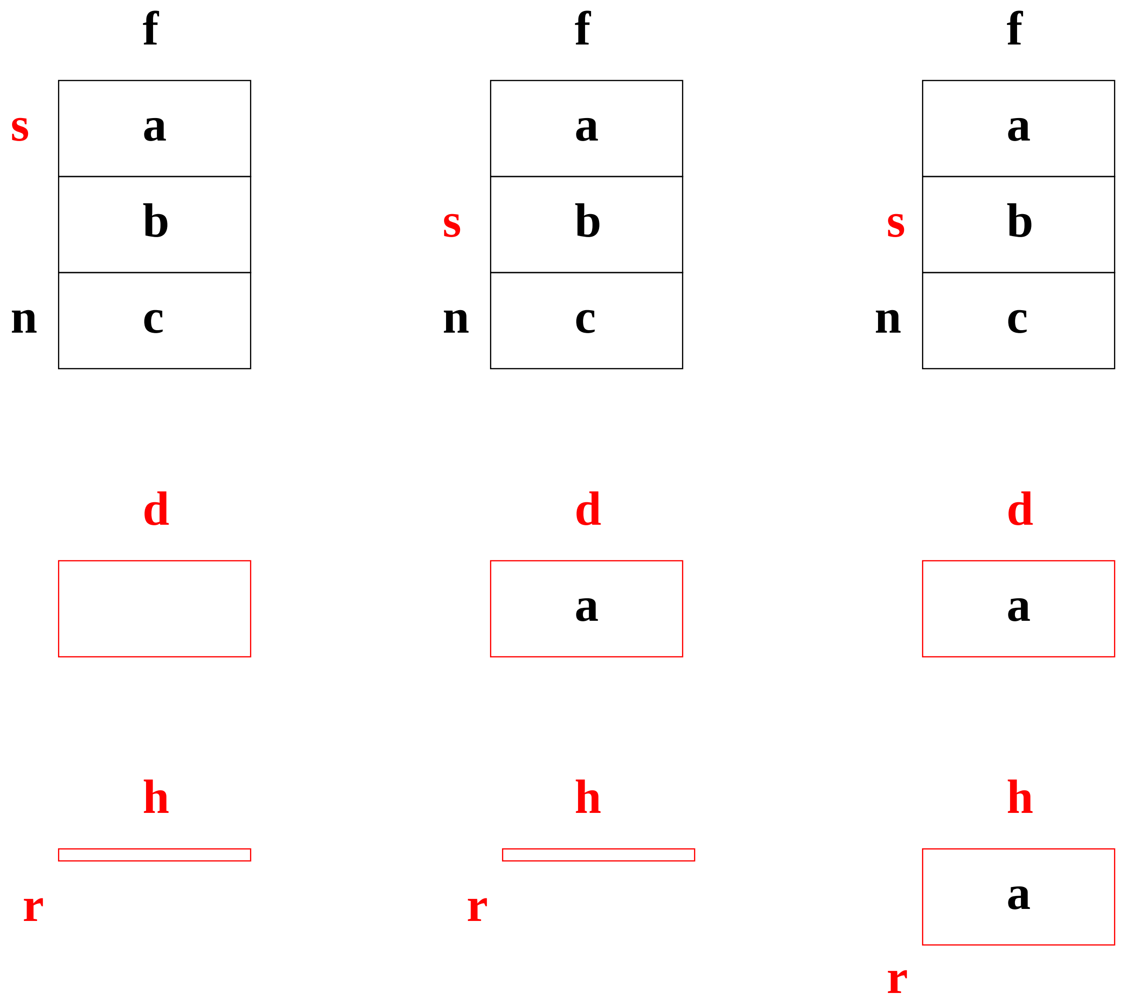


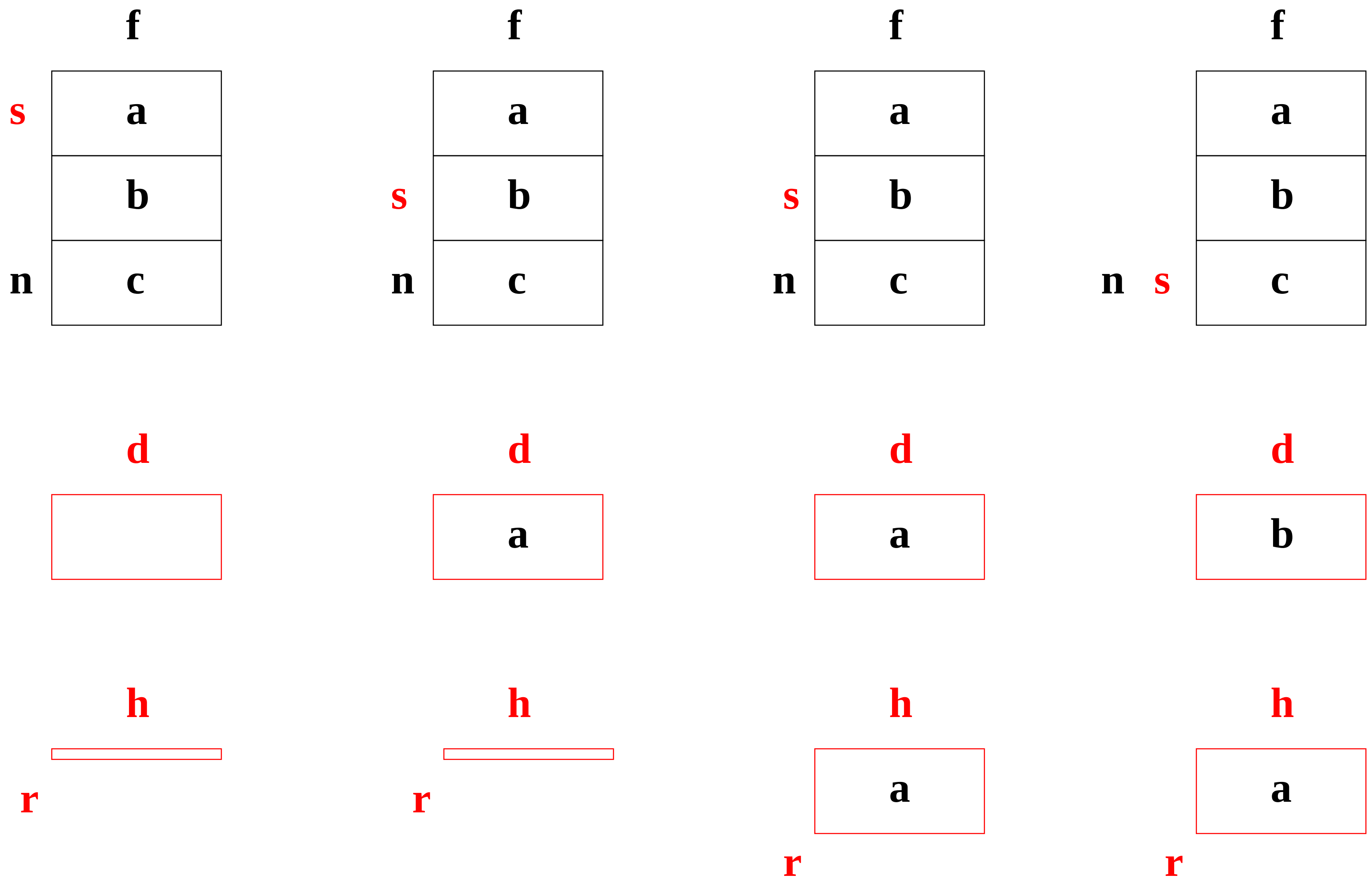
Initial Situation




f
s
a
b
n
c
d
h
r
f
a
s
b
n
c
d
a
h
r

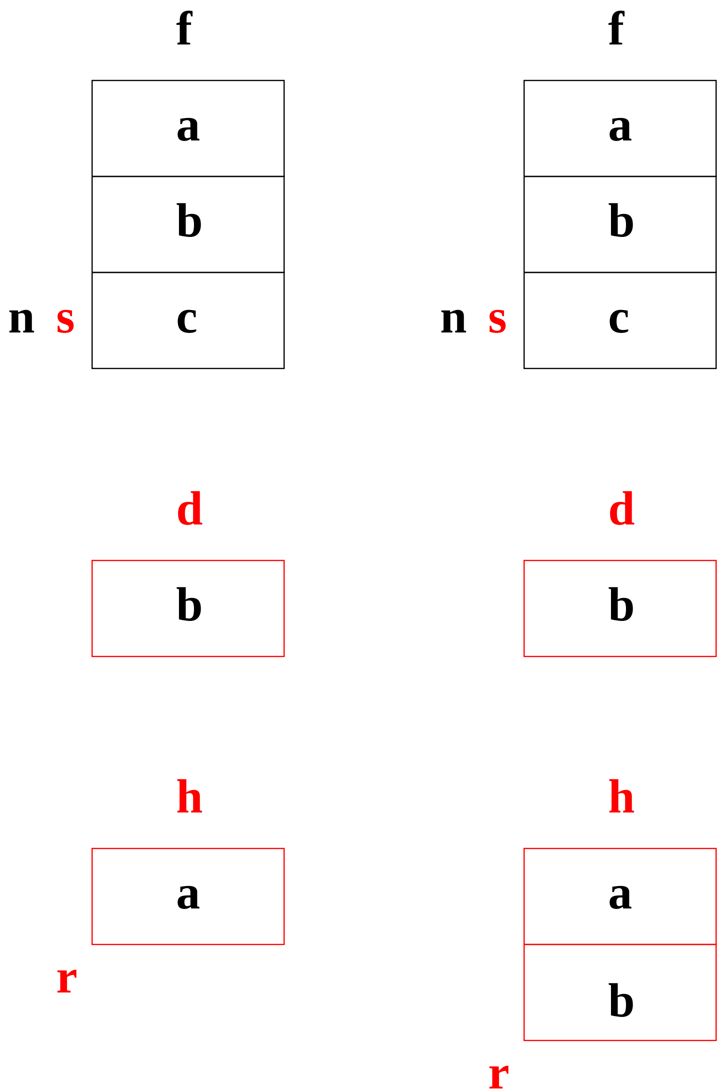
Receive

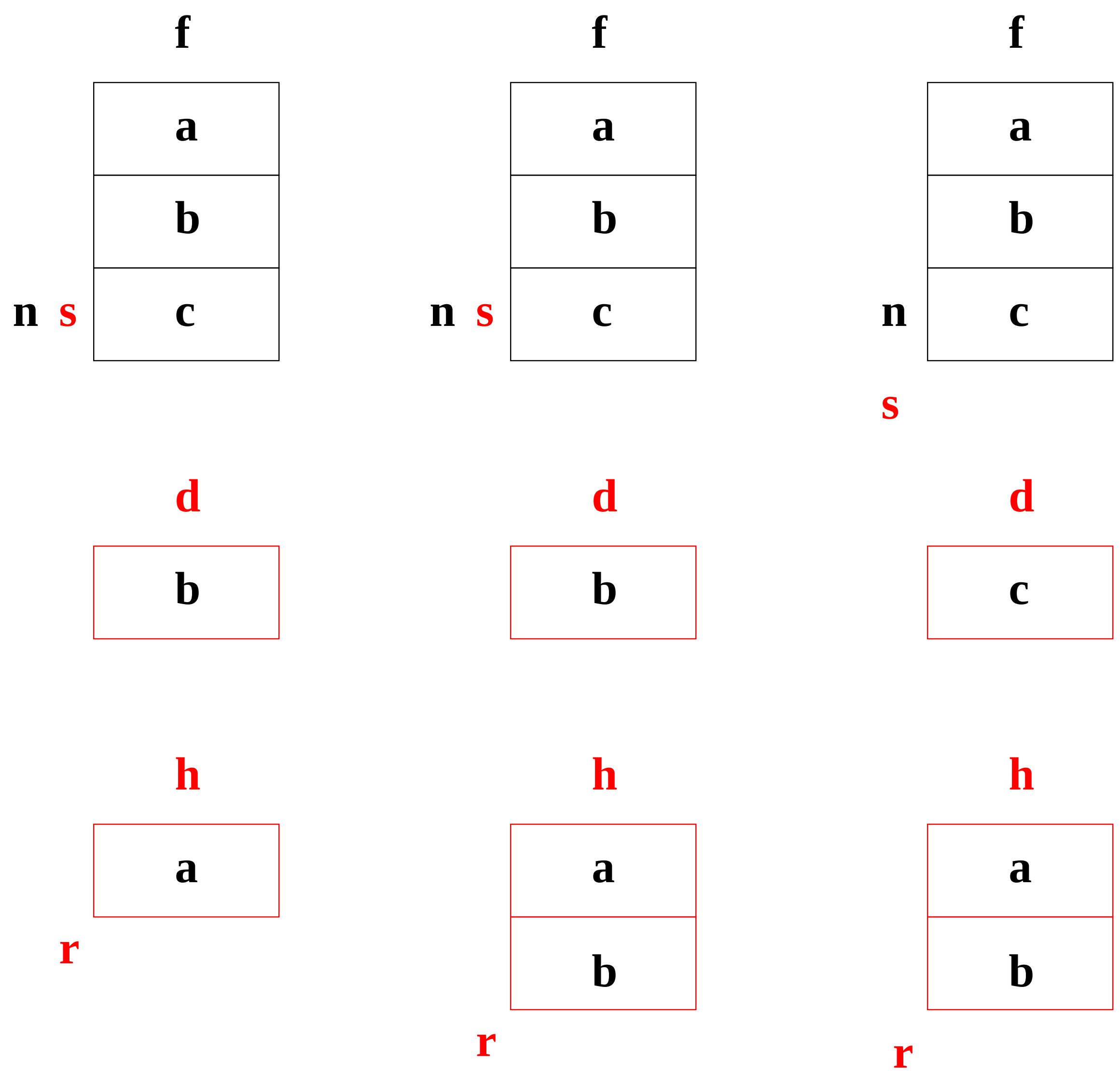




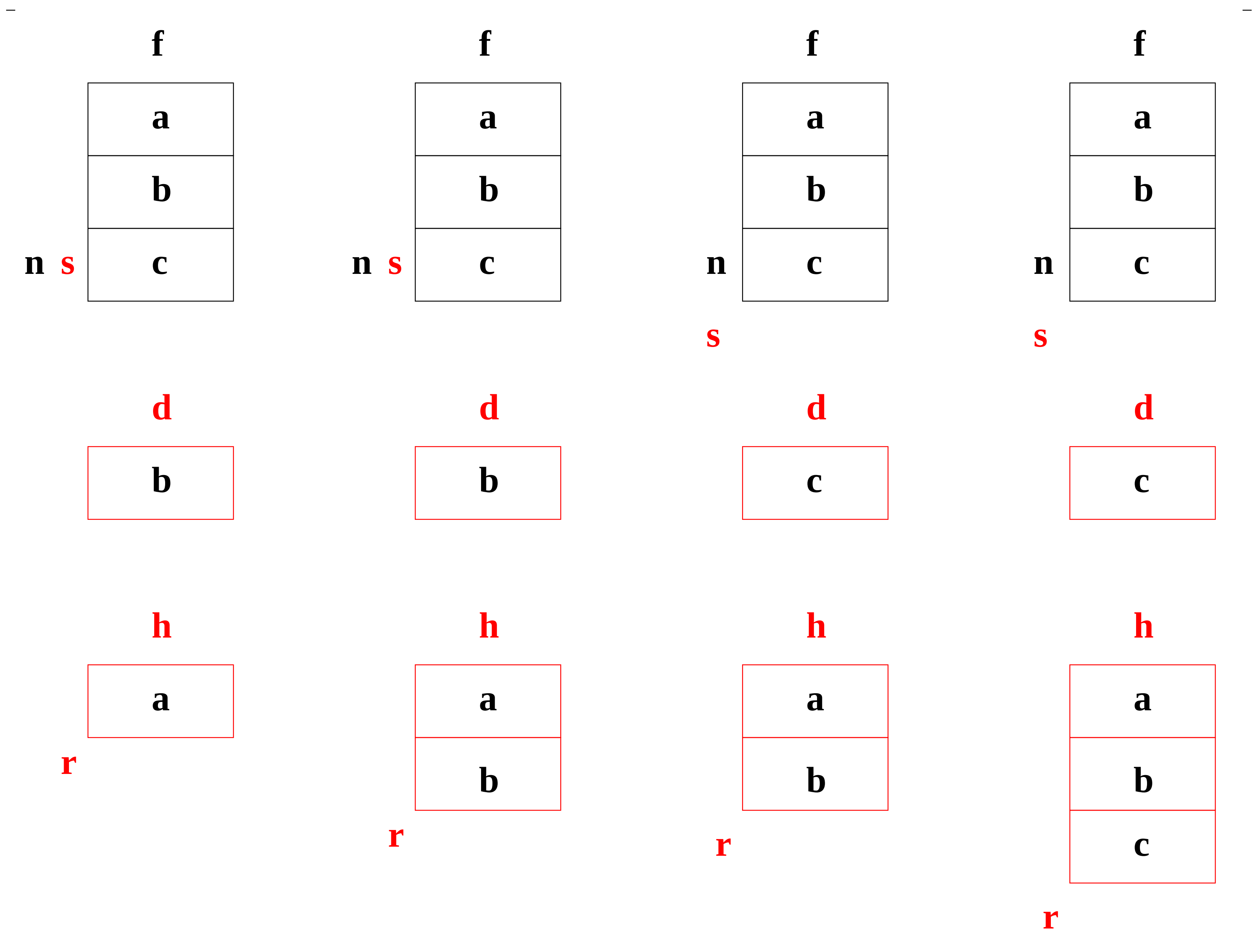
f
s a
b
n c
d
h
r
f
a
s b
n c
d
a
h
r
f
a
s b
n c
d
a
h
a
r
f
a
b
n s c
d
b
h
a
r

Receive

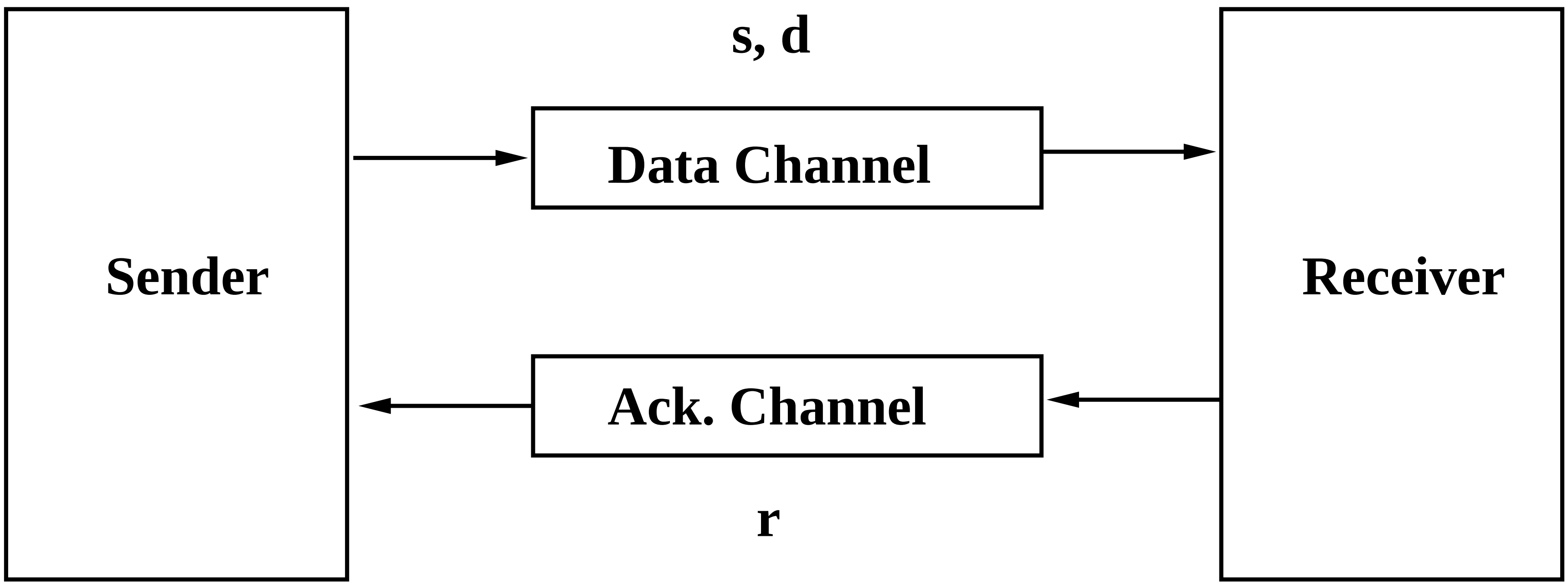



f
a
b
n s c
d
b
h
a
r
f
a
b
n s c
d
b
h
a
b
r
f
a
b
n c
s
d
c
h
a
b
r

Receive



Sender and Receiver



- We introduce an additional variable s, and a data item d

variables: b, h, r, s, d	**inv2_1:** $s \leq n + 1$ **inv2_2:** $s \in r \,..\, r + 1$ **inv2_3:** $s = r + 1 \Rightarrow d = f(r)$

init
$$b := \mathrm{FALSE}$$
$$h := \varnothing$$
$$r := 1$$
$$s := 1$$
$$d :\in D$$

send
when
$$s = r$$
$$s \neq n + 1$$
then
$$d := f(s)$$
$$s := s + 1$$
end

receive
when
$$s = r + 1$$
then
$$h := h \cup \{r \mapsto d\}$$
$$r := r + 1$$
end

final
when
$$b = \mathrm{FALSE}$$
$$r = n + 1$$
then
$$b := \mathrm{TRUE}$$
end

What is to be Proved

- Event init refines its abstraction

- Event final refines its abstraction

- Event receive refines its abstraction

- Event send refines skip

- Event send does not diverge

- Relative deadlock freeness

Refinement Strategy

- Initial model: The file is transmitted in one shot (FUN1 and FUN2)

- First refinement: The file is transmitted gradually (FUN3)

- Second refinement: The two agents are separated

- Third refinement: Towards an implementation

Some Ideas for a Third Refinement?

send
when
$s = r$
$s \neq n + 1$
then
$d := f(s)$
$s := s + 1$
end

receive
when
$s = r + 1$
then
$h := h \cup \{r \mapsto d\}$
$r := r + 1$
end

inv2_2: $s \in r \,..\, r + 1$

Some Ideas for a Third Refinement?

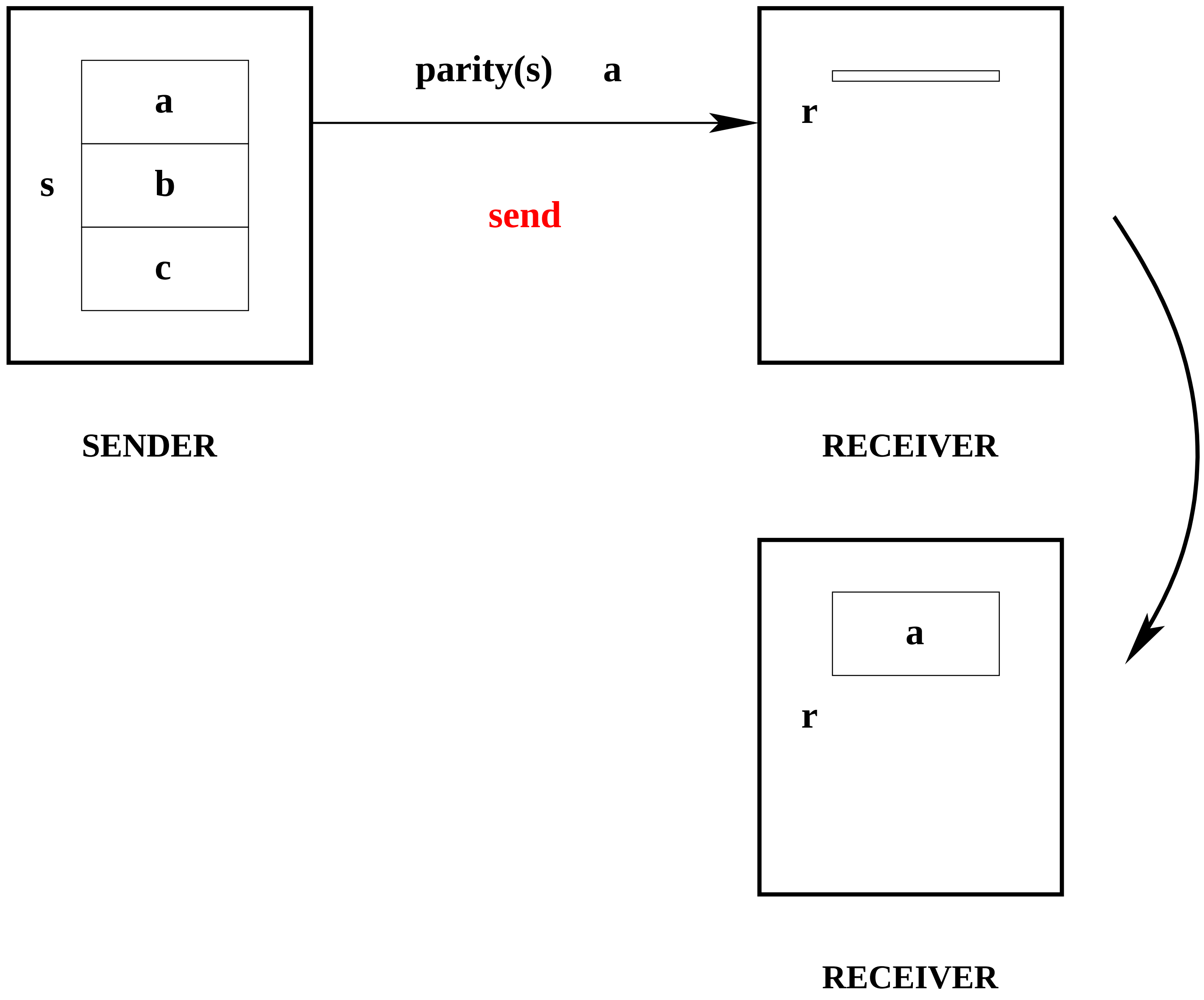
send
when
$s = r$
$s \neq n + 1$
then
$d := f(s)$
$s := s + 1$
end

receive
when
$s = r + 1$
then
$h := h \cup \{r \mapsto d\}$
$r := r + 1$
end

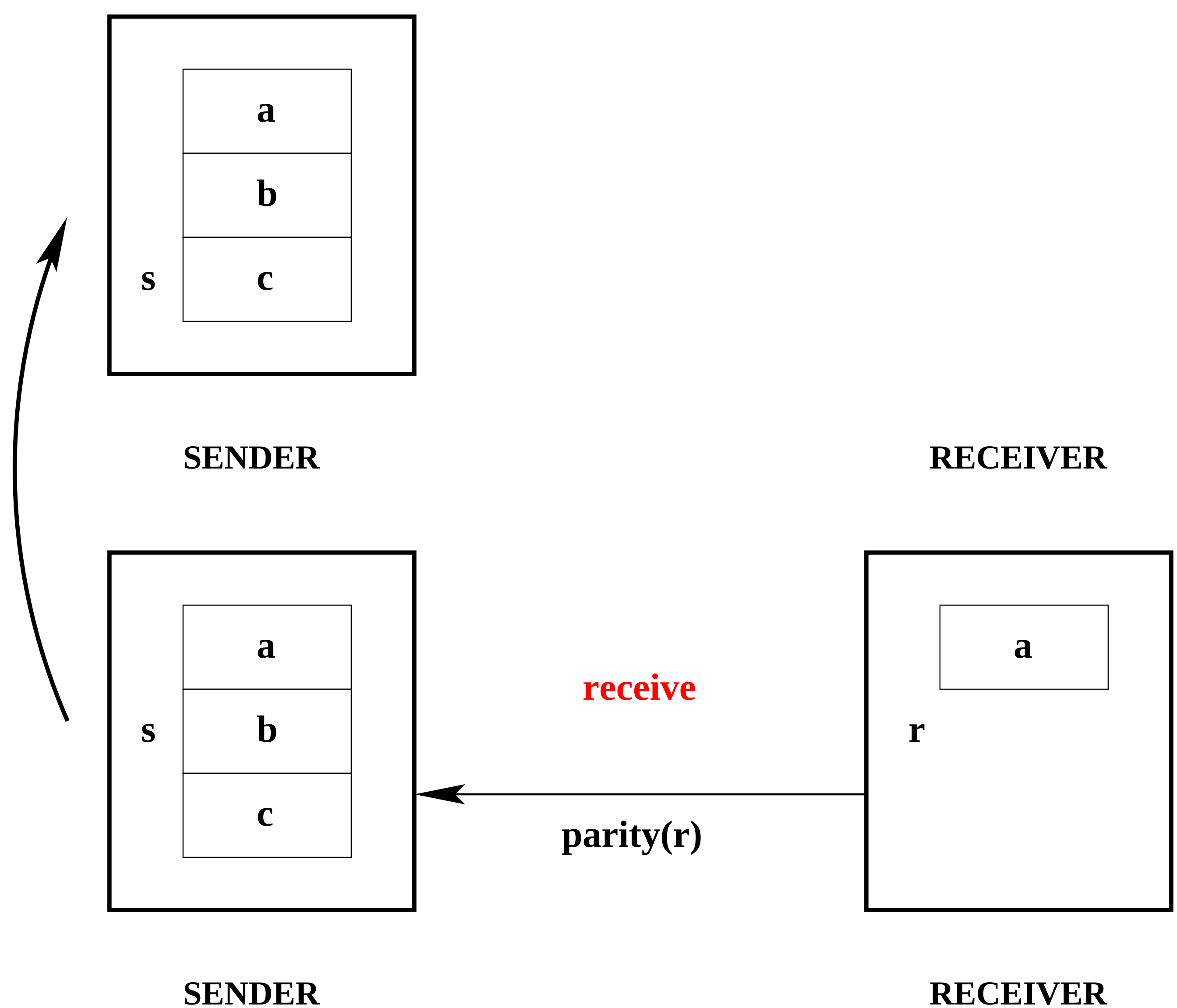
inv2_2: $s \in r \,..\, r + 1$

- In order to compare r and s, it is sufficient to compare their parities

Third Refinement: Sending Parities (1)



Third Refinement: Sending Parities (2)



axm3_1: $parity \in \mathbb{N} \rightarrow \{0, 1\}$

axm3_2: $parity(0) = 0$

axm3_3: $\forall x \cdot x \in \mathbb{N} \Rightarrow parity(x+1) = 1 - parity(x)$

thm3_1:
$$\forall x, y \cdot \begin{array}{l} x \in \mathbb{N} \\ y \in \mathbb{N} \\ x \in y \mathrel{..} y+1 \\ parity(x) = parity(y) \end{array} \Rightarrow x = y$$

variables: b
h
r
s
d
p
q

inv3_1: $p = parity(s)$

inv3_2: $q = parity(r)$

init
$b := \mathrm{FALSE}$
$h := \varnothing$
$r := 1$
$s := 1$
$d :\in D$
$p := 1$
$q := 1$

final
when
$b = \mathrm{FALSE}$
$r = n + 1$
then
$b := \mathrm{TRUE}$
end

send
when
$p = q$
$s \neq n + 1$
then
$d := f(s)$
$s := s + 1$
$p := 1 - p$
end

receive
when
$p \neq q$
then
$h := h \cup \{r \mapsto d\}$
$r := r + 1$
$q := 1 - q$
end

Third Refinement: Proofs

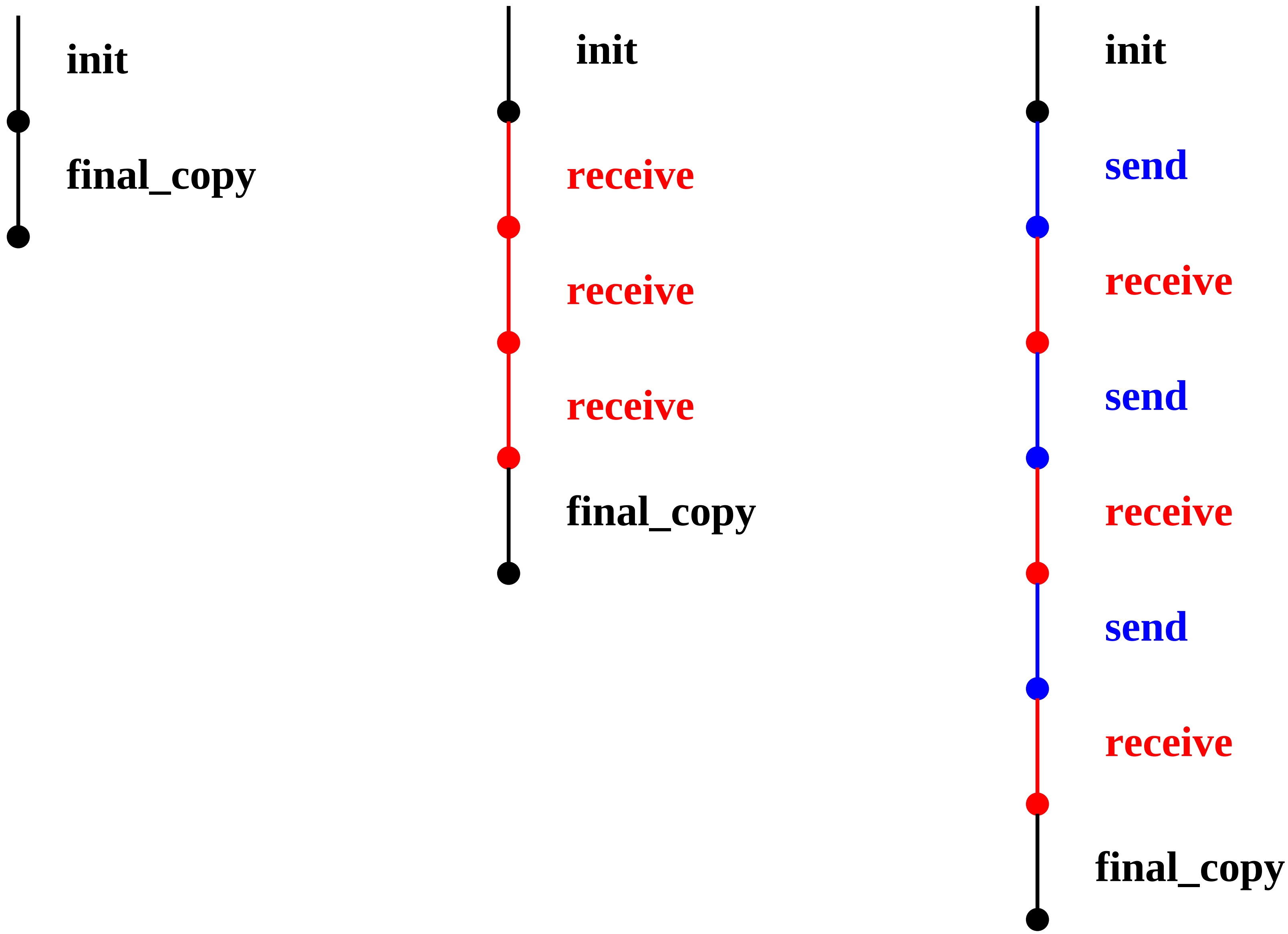
- The proofs are left as exercises

Summary of Proofs with the Tool

	Total	Interactive
Initial Model	6	0
1st Refinement	13	0
2nd Refinement	15	0
3rd Refinement	8	5
Total	42	5

- More mathematical conventions
- How to write a model
- What kind of things we have to prove
- How the proof can help finding invariants
- Many things can be done by tools
- A small theory of parities

Gradual Observation of the Intended System



Reminder of Mathematical Conventions (1)

$x \in S$	Set membership operator
$\mathbb{N}$	set of Natural Numbers: $\{0, 1, 2, 3, \ldots\}$
$a \mathinner{\ldotp\ldotp} b$	Interval from a to b: $\{a, a+1, \ldots, b\}$ (empty when $b < a$)
$a \mapsto b$	pair constructing operator
$S \times T$	Cartesian product operator
$S \subseteq T$	set inclusion operator
$\mathbb{P}(S)$	power set operator

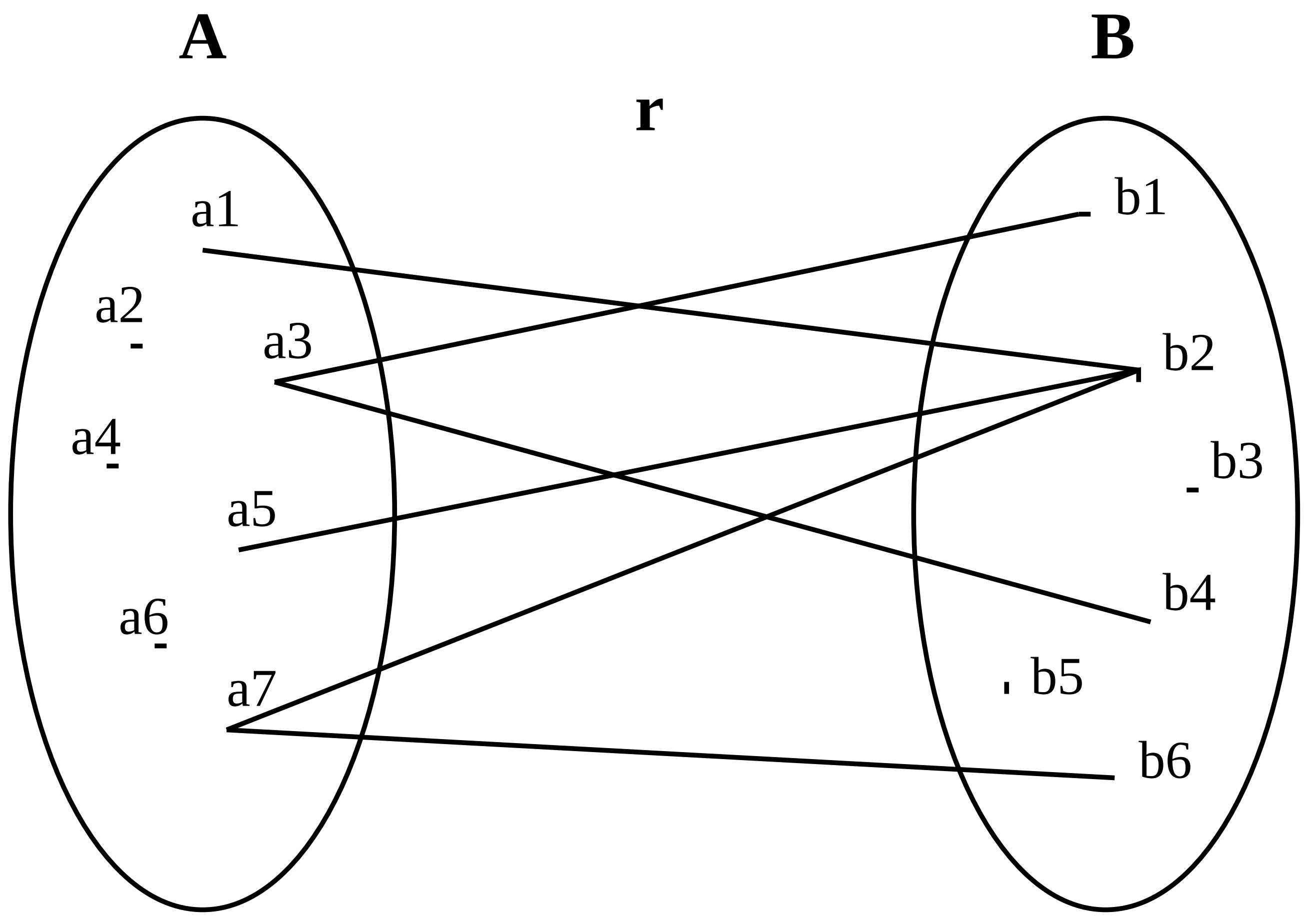
Reminder of Mathematical Conventions (2)

$S \leftrightarrow T$	Set of binary relations from S to T
$S \rightarrow T$	Set of total functions from S to T
$S \nrightarrow T$	Set of partial functions from S to T
$\mathrm{dom}(r)$	Domain of a relation r
$\mathrm{ran}(r)$	Range of a relation r

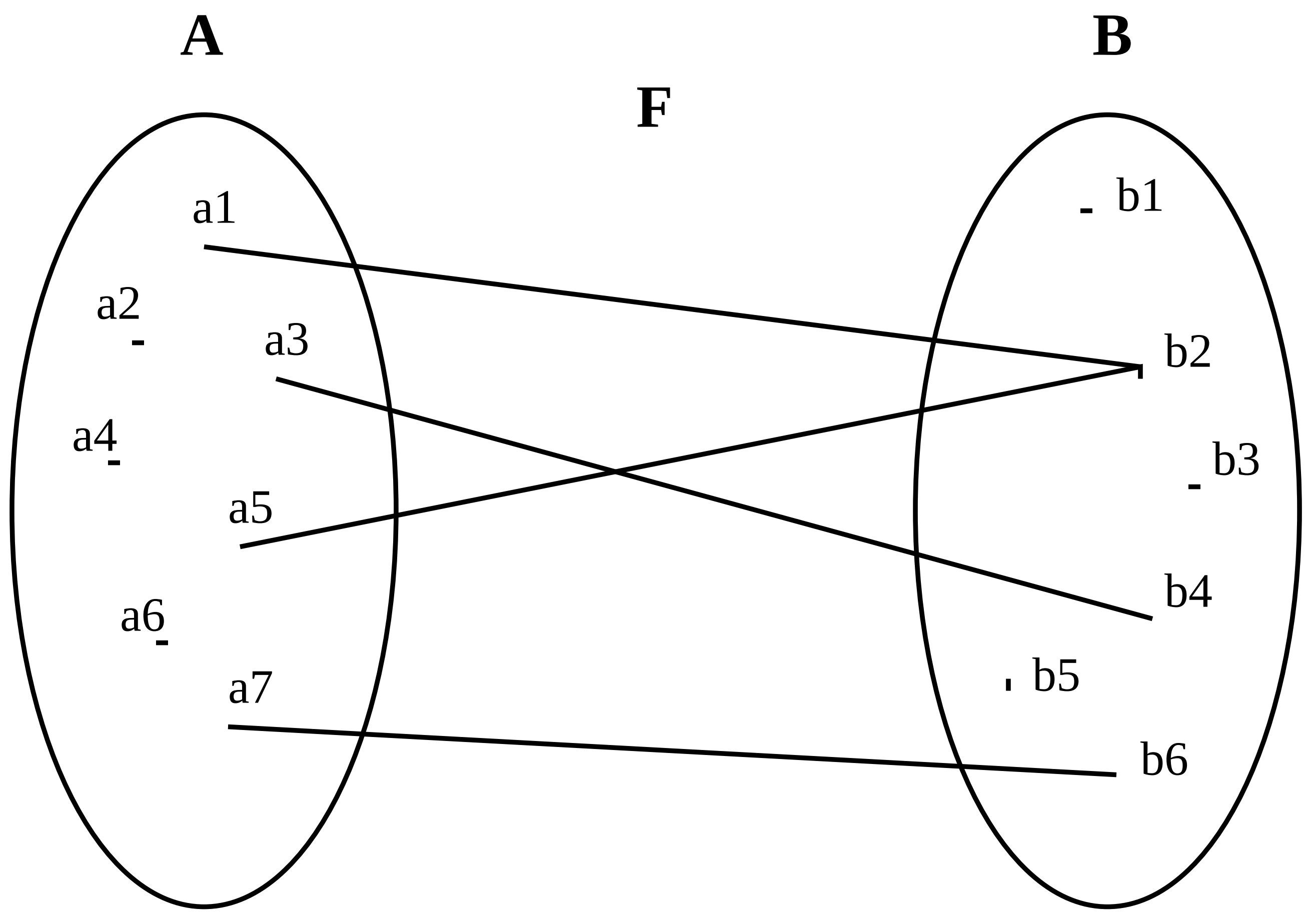
Reminder of Mathematical Conventions (3)

$s \lhd r$	domain restriction operator
$s ⩤ r$	domain subtraction operator
$r \rhd t$	range restriction operator
$r ⩥ t$	range subtraction operator

A Binary Relation r from a Set A to a Set B

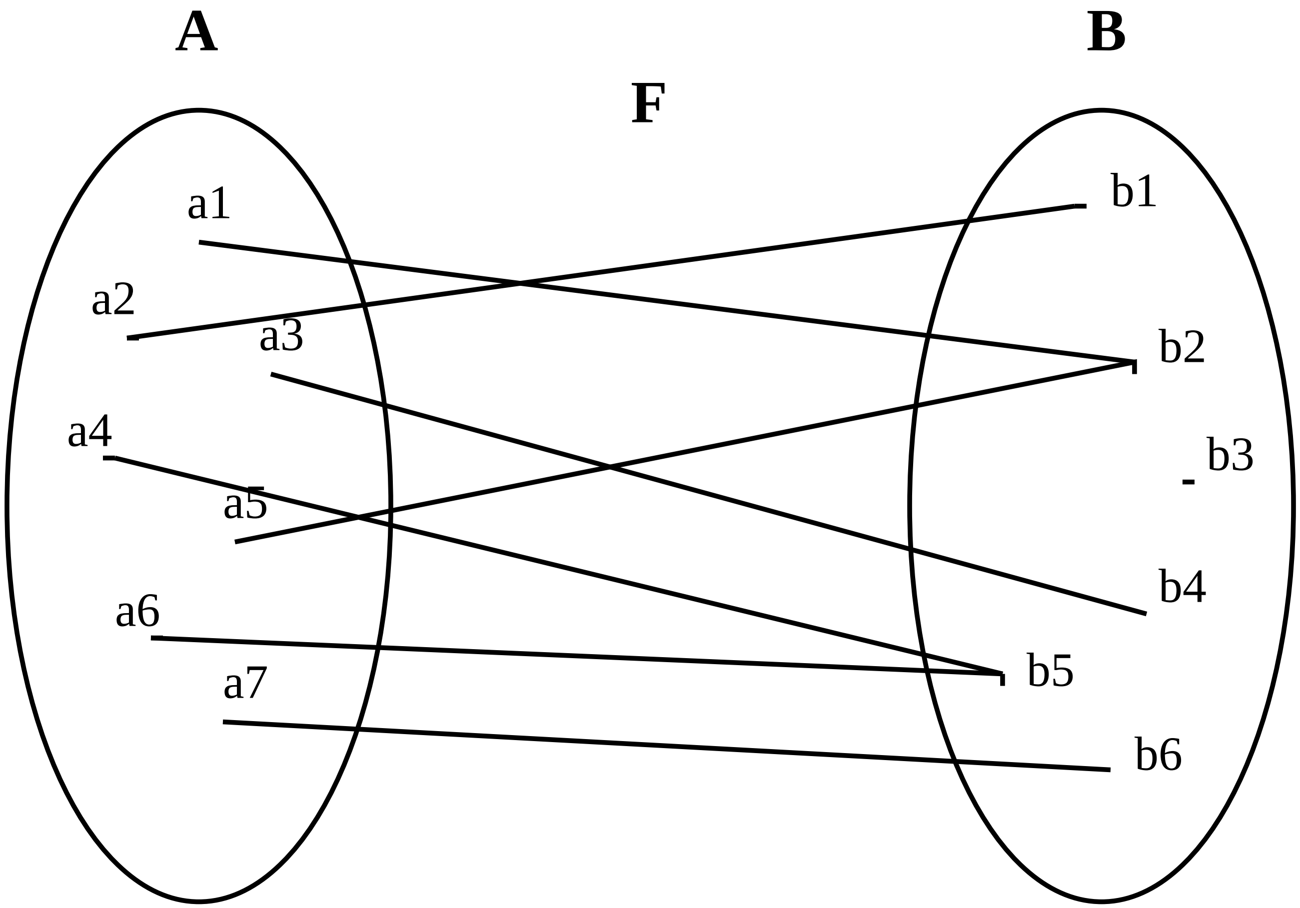


A Partial Function F from a Set A to a Set B

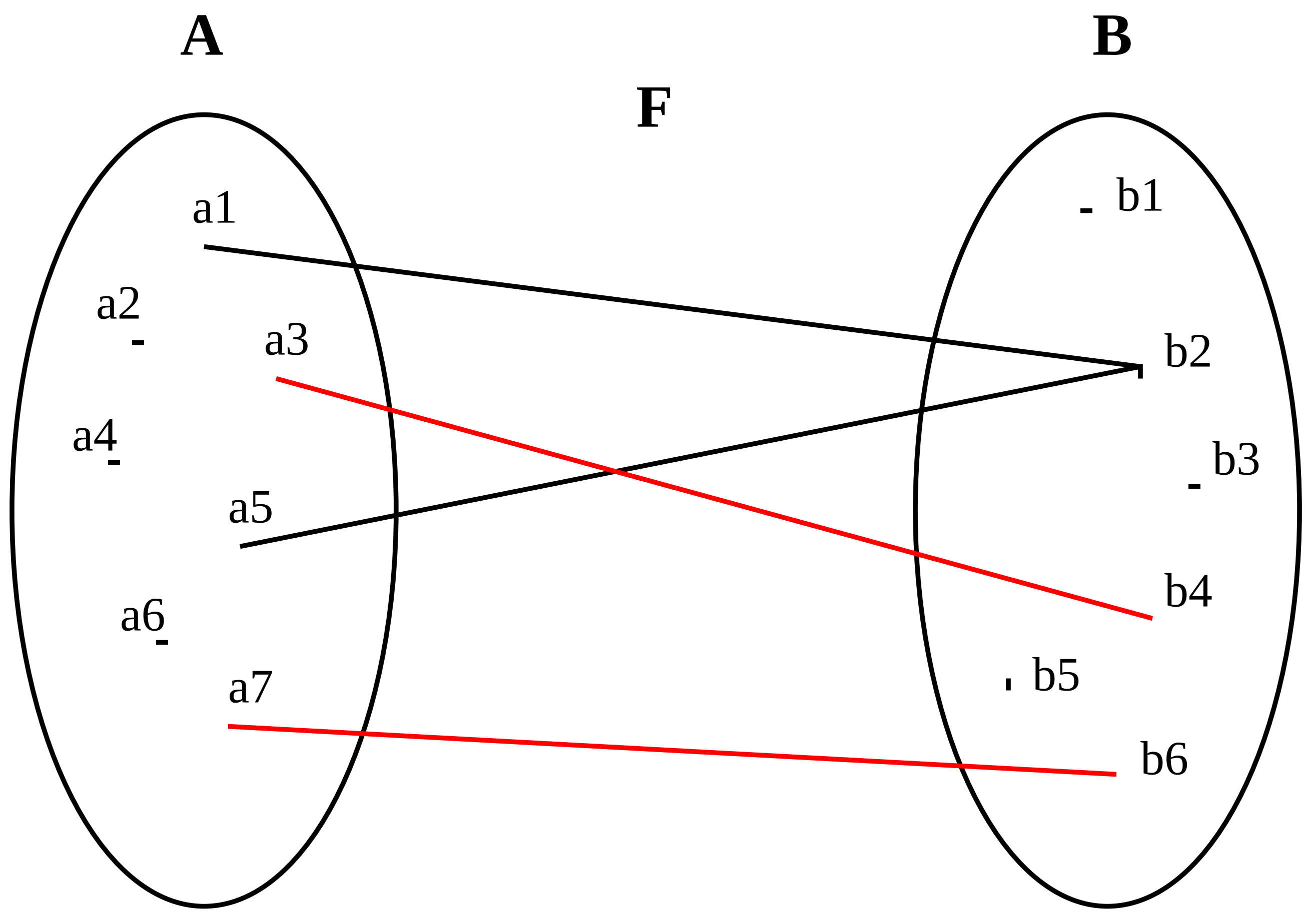


$$
\begin{aligned}
F &= \{a1 \mapsto b2,\ \ a3 \mapsto b4,\ \ a5 \mapsto b2,\ \ a7 \mapsto b6\} \\
\mathrm{dom}\,(F) &= \{a1,\ \ a3,\ \ a5,\ \ a7\} \\
\mathrm{ran}\,(F) &= \{b2,\ \ b4,\ \ b6\}
\end{aligned}
$$

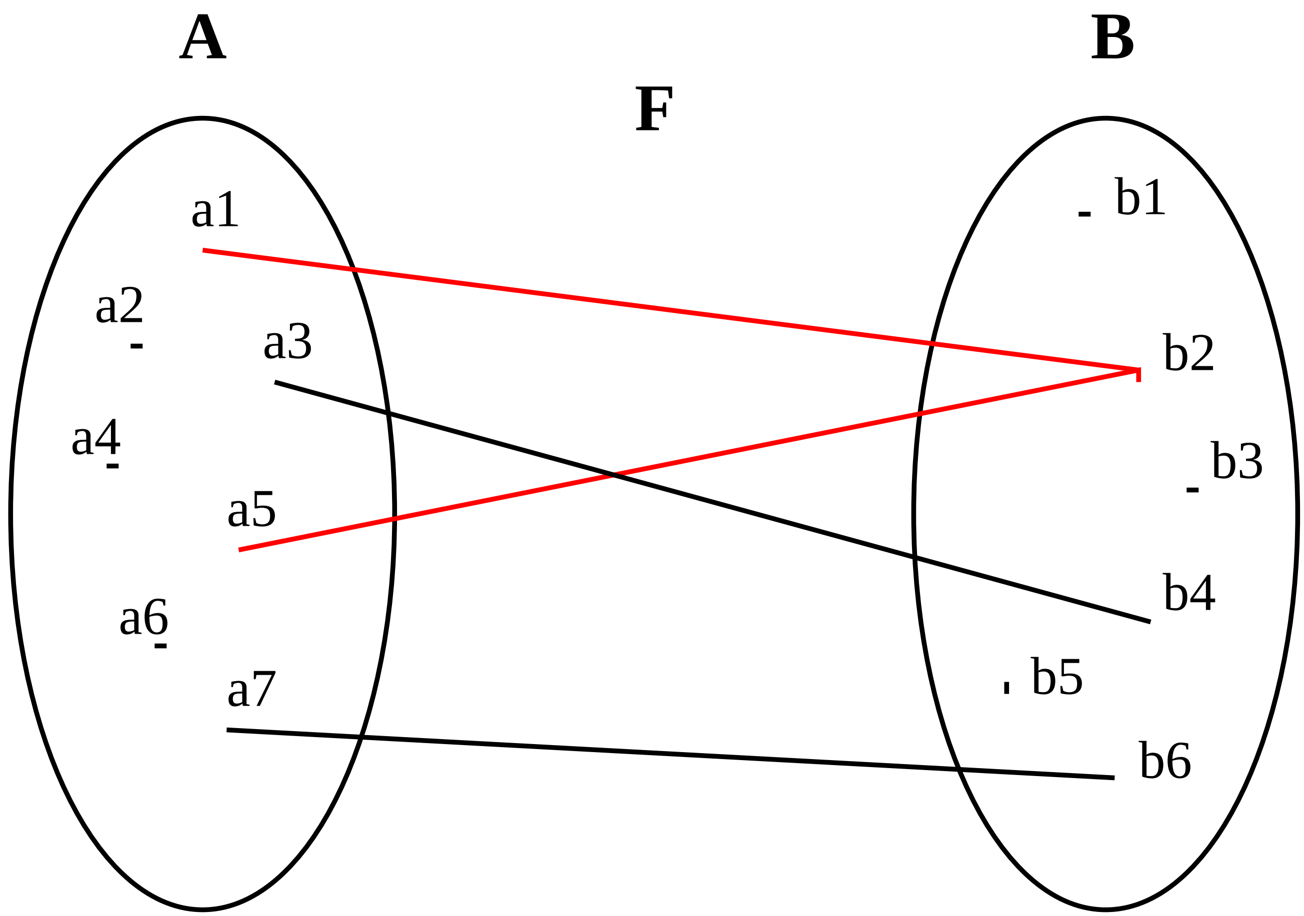
A Total Function F from a Set A to a Set B



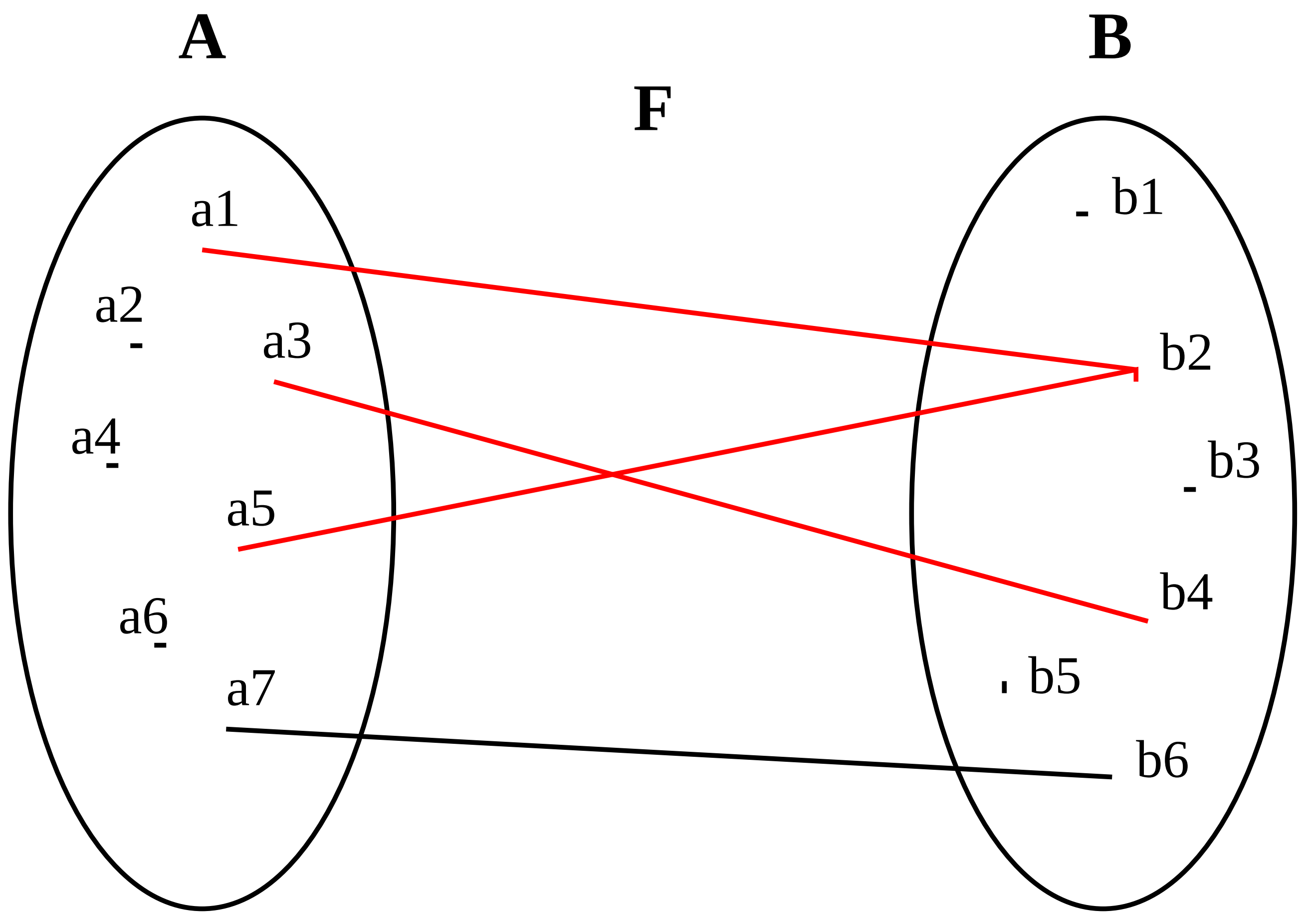
$$\mathsf{dom}\,(F) \;=\; A$$



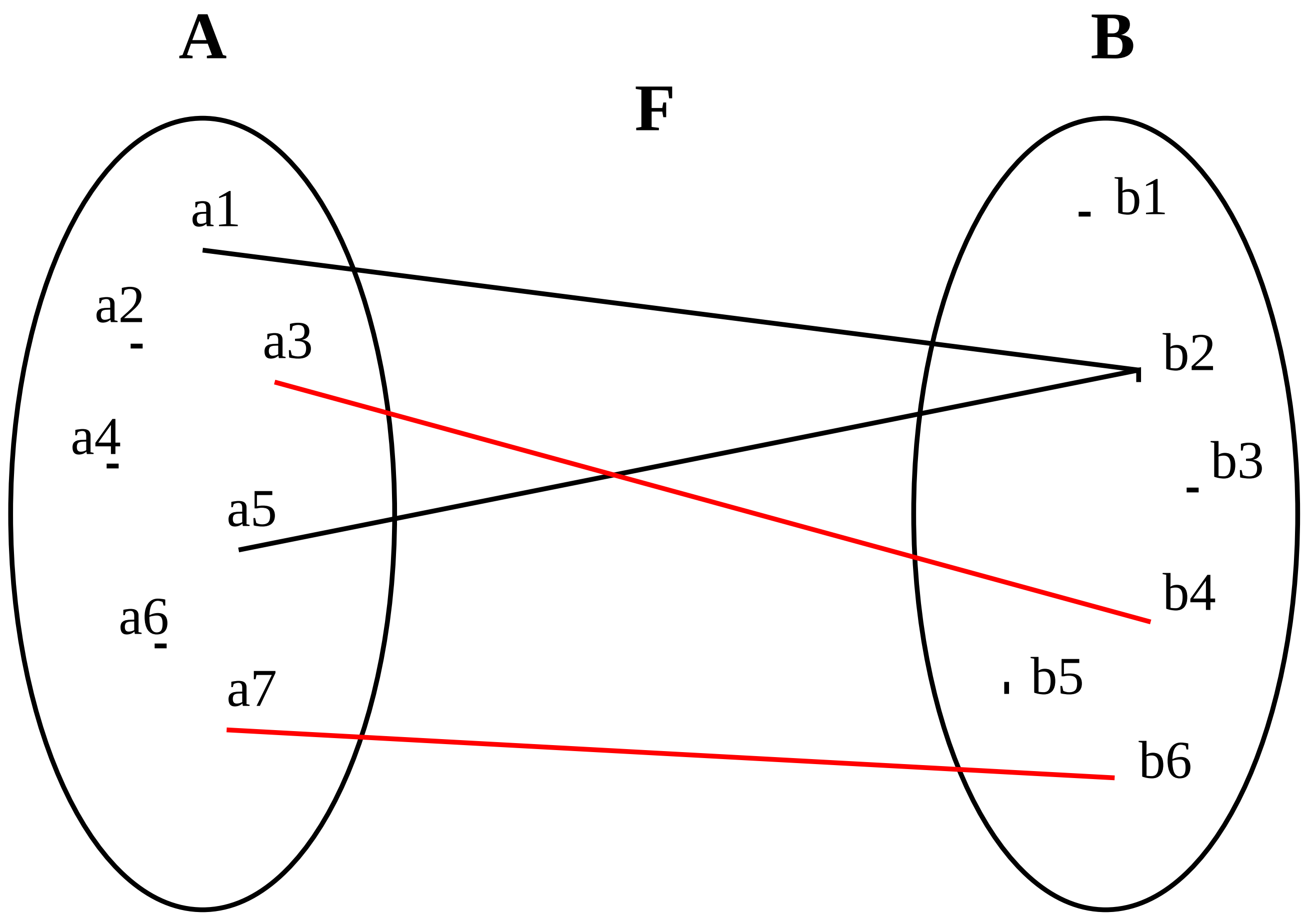
$$\{a3,\ a7\} \lhd F$$



$$\{a3,\ a7\} \lhd F$$



$$F \rhd \{b2, b4\}$$



$$F \unrhd \{b2\}$$

Structure of a Model

- List of Sets (identifiers)

- List of Constants (identifiers)

- List of Axioms (predicates built on sets and constants)

- List of Variables (identifiers)

- List of Invariants (predicates built on sets, constants, and variables)

- List of Events

Summary of the File Transfer Protocol Initial Model

sets: D

constants: n
f

axm0_1: $0 < n$

axm0_2: $f \in 1 \,..\, n \rightarrow D$

variables: g
b

inv0_1: $g \in 1 \,..\, n \nrightarrow D$

inv0_2: $b = \text{FALSE} \Rightarrow g = \varnothing$

inv0_3: $b = \text{TRUE} \Rightarrow g = f$

init
$g := \varnothing$
$b := \text{FALSE}$

final
when
$b = \text{FALSE}$
then
$g := f$
$b := \text{TRUE}$
end

variables: b, h, r

inv1_1: $r \in 1\,..\,n+1$

inv1_2: $h = (1\,..\,r-1) \lhd f$

inv1_3: $b = \mathrm{TRUE} \Rightarrow r = n+1$

thm1_1: $b = \mathrm{TRUE} \Rightarrow h = g$

variant1: $n + 1 - r$

init
$b := \mathrm{FALSE}$
$h := \varnothing$
$r := 1$

receive
when
$r \leq n$
then
$h := h \cup \{r \mapsto f(r)\}$
$r := r + 1$
end

final
when
$b = \mathrm{FALSE}$
$r = n + 1$
then
$b := \mathrm{TRUE}$
end

- Variable g has disappeared: it is not satisfactory

Introducing Anticipated Events

- We want to keep the variable g in the first refinement

- That seems impossible since event receive has to refine skip

- For this, we introduce the notion of anticipated event

- Such an event will be later proved to be conveergent

Initial Model with an anticipated event

sets: D

constants: n
f

axm0_1: $0 < n$

axm0_2: $f \in 1\,..\,n \rightarrow D$

variables: g

inv0_1: $g \in 1\,..\,n \rightarrow\!\!\!\!\!+ D$

init
  $g := \varnothing$

receive
  status
    anticipated
  when
    $g \neq f$
  then
    $g :\in 1\,..\,n \rightarrow\!\!\!\!\!+ D$
  end

final
  when
    $g = f$
  then
    skip
  end

- Event receive is highly non-deterministic.

- Notice the event final

- Variable b is not useful anymore

variables: g
r

inv1_1: $r \in 1 \,..\, n+1$

inv1_2: $g = (1 \,..\, r-1) \lhd f$

variant1: $n+1-r$

init
$g := \varnothing$
$r := 1$

receive
status
convergent
when
$r \neq n+1$
then
$g(r) := f(r)$
$r := r+1$
end

final
when
$r = n+1$
then
skip
end

Note: the event receive works now with variable g