Formal Approaches for Detecting Feature Interactions, Their Experimental Results, and Application to VoIP

T. Yoneda, S. Kawauchi,

J. Yoshida and T. Ohta

SOKA University

Contents

1. Background and Problems
2. Problems and Their Solutions
    - Terminal Assignments
    - Reachability Test
    - Static Detection Algorithm
3. Implementation and Evaluations
4. Application to VoIP

Background

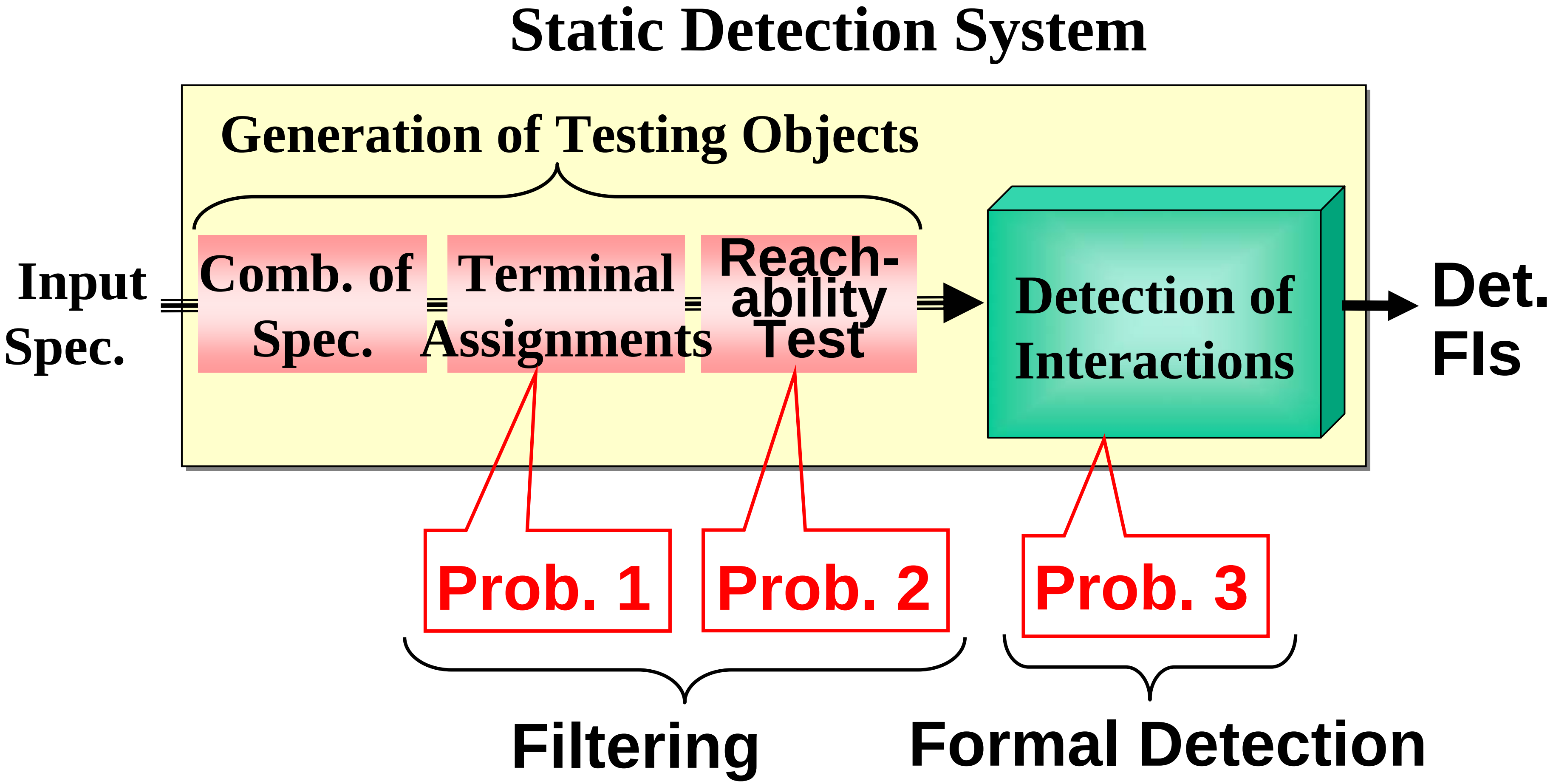
- **Dynamic Detection: Detecting interactions by executing service specifications.**

  Explosion in computation time for detecting feature interactions

- **Static Detection: Detecting interactions solely by analyzing service specifications**

  Coverage and Redundancy in detecting feature interactions

Problems in Static Detection



Contents

1. Background and Problems
2. Problems and Their Solutions
   - Terminal Assignments
   - Reachability Test
   - Static Detection Algorithm
3. Implementation and Evaluations
4. Application to VoIP

Specification Description Language
STR:State Transition Rule

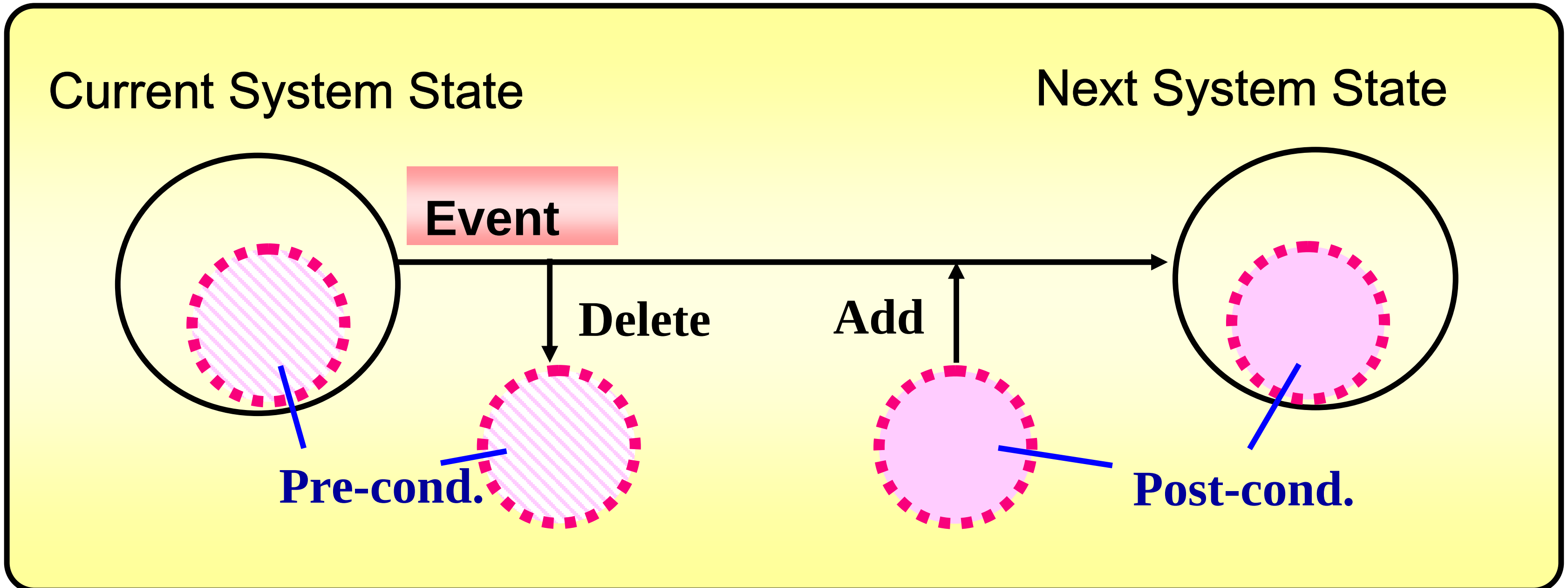
Specification is represented as a set of STR rules

Form : Pre-cond. Event: Post-cond.

Pre-cond. and Post-cond. are represented as a set of primitives.

Rule application : Precondition exists in the current system state

State change:

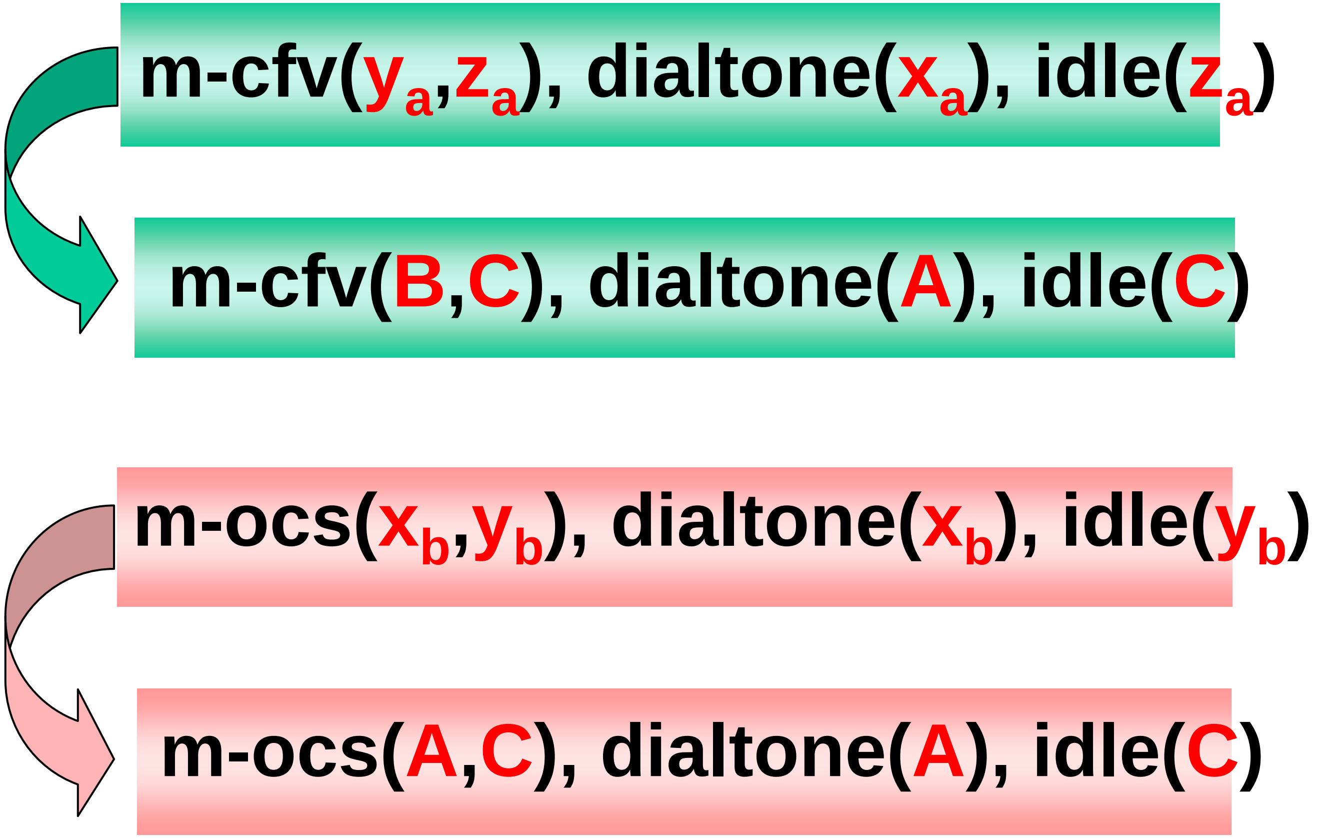


Terminal Assignments

Terminal is described as a variable in rules.

To detect FIs, real terminals are assigned to variables.

Ex. :



m-cfv(y_a,z_a), dialtone(x_a), idle(z_a)

m-cfv(B,C), dialtone(A), idle(C)

m-ocs(x_b,y_b), dialtone(x_b), idle(y_b)

m-ocs(A,C), dialtone(A), idle(C)

Problems so far

It was not clear
how to assign real terminals to terminal variables.

Explosive computation time with all assignments
Low coverage with reduced assignments

Proposal of terminal assignment method, where
unnecessary terminal assignments are deleted.

Basic Idea for Terminal Assignment

No terminals belong to both services: no feature interactions

If a terminal belongs to both services, feature interactions may occur.

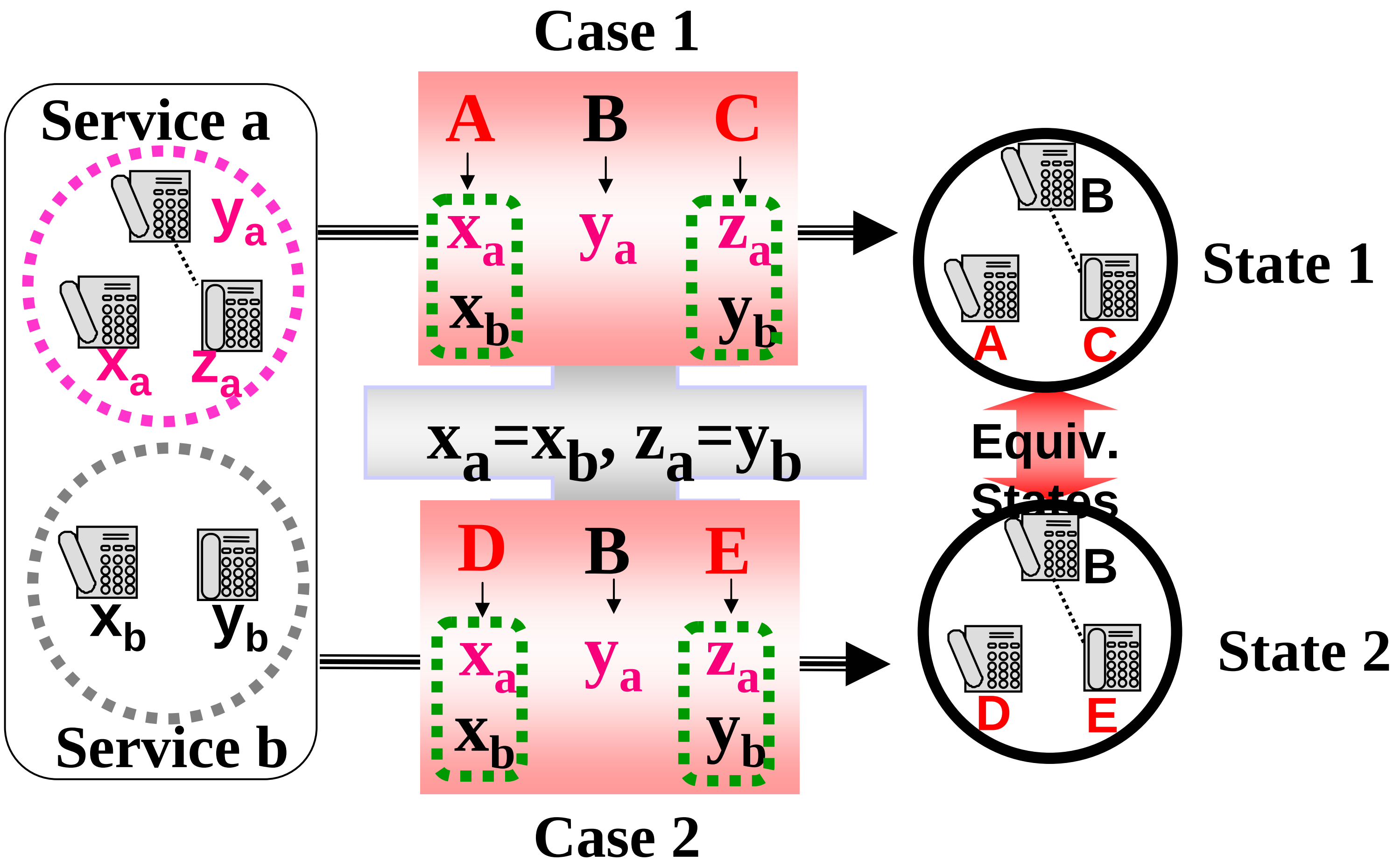
A terminal belongs to both services:

xa for service A, xb for service B   xa=xb=terminal P

Combination of variables: a set of pairs of variables to which the same real terminals are assigned

Different terminal assignments to the same combination of variable gives equivalent states, the same state with different terminal names.

Equivalent States



Combination of variables are the same.
Terminal assignments are different.

Basic Idea for Terminal Assignment

No terminals belong to both services: no feature interactions

If a terminal belongs to both services, feature interactions may occur.

A terminal belongs to both services:

xa for service A, xb for service B   xa=xb=terminal P

Combination of variables: a set of pairs of variables to which the same real terminals are assigned

Different terminal assignments to the same combination of variable gives equivalent states, the same state with different terminal names.

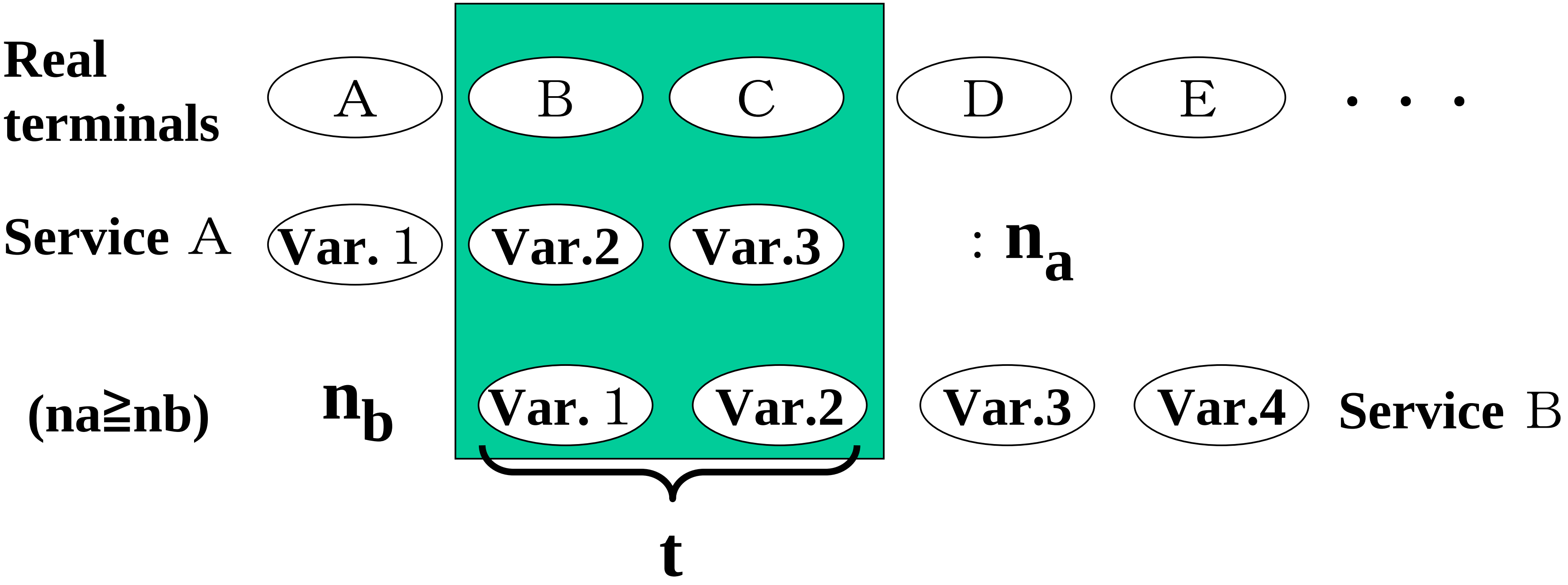
Interactions caused in equivalent states are equivalent interactions.

⇓

One terminal assignment to one combination of variables.

Consider only different combination of variables.

The number of combinations of variables



Real terminals A B C D E . . .

Service A Var. 1 Var.2 Var.3 : n_a

(na≧nb) n_b Var. 1 Var.2 Var.3 Var.4 Service B

t

$$N = \sum_{t=1}^{t=n_a} {}_{n_a}C_t \times {}_{n_b}P_t$$

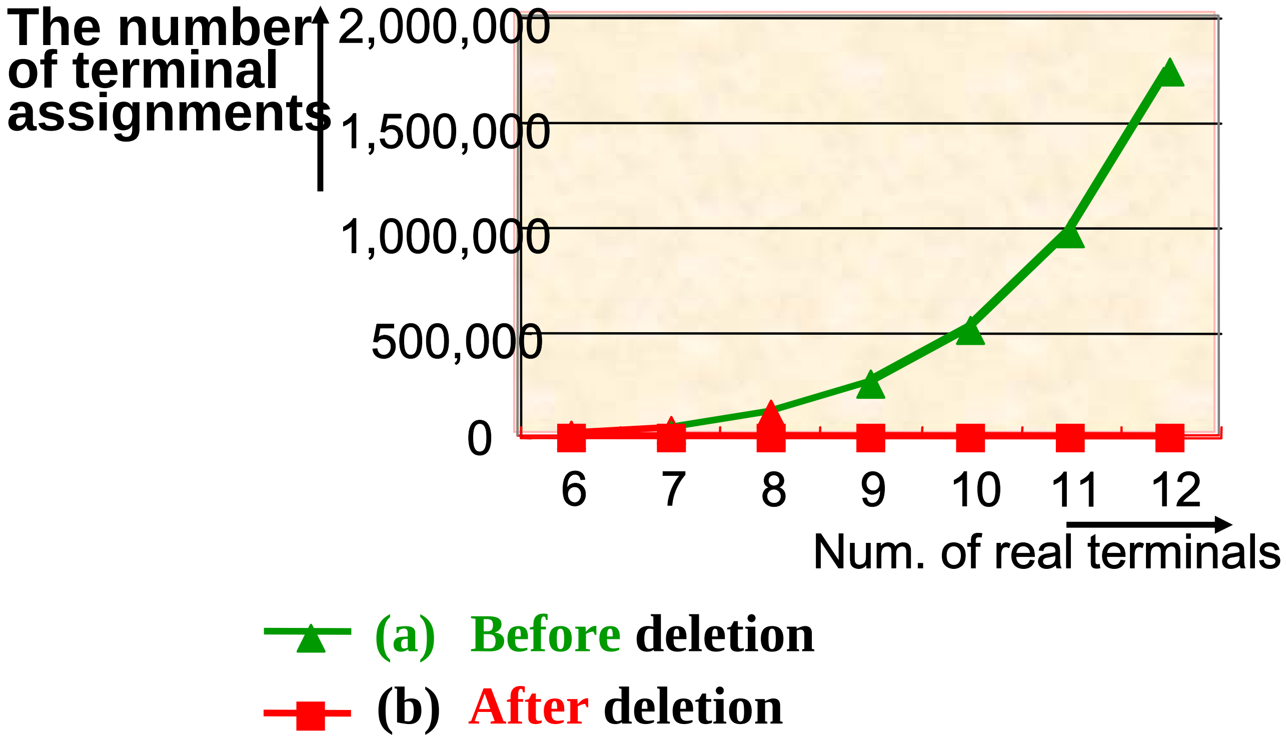
When $n_a = n_b = 3$, N=33

The number of all terminal assignments: 14400

${}_{n_T}P_{n_a} \times {}_{n_T}P_{n_b}$ Here $n_T = n_a + n_b$

Effects of Deleting Equiv. Term.

The number of terminal assignments for a service pair which have 3 term. variables.



Num of real ter.	6	7	8	9	10	11	12	. . .	100
(a)	14,400	44,100	112,896	254,016	518,400	980,100	1,742,400	. . .	941,288,040,000
(b)	33	33	33	33	33	33	33		33

Problem 2

Reachability Test

New Method

Conventional

Generating states

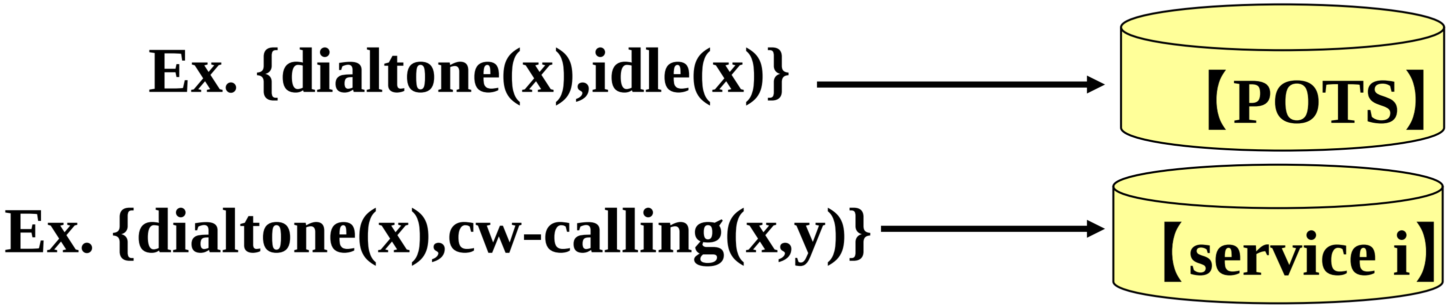
P-invariant in Petri-Net

→ Require much time

New Proposal

Using knowledge which can be obtained easily

Illegal Combinations of Primitives

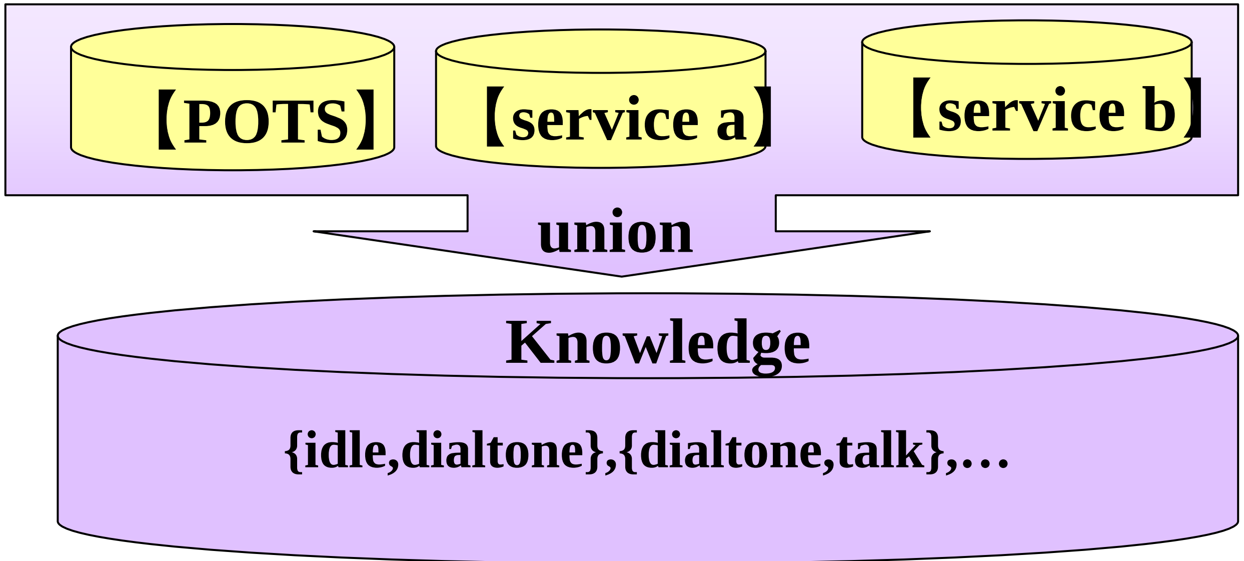


Ex. {dialtone(x),idle(x)} → 【POTS】

Ex. {dialtone(x),cw-calling(x,y)} → 【service i】

considering only service i and POTS

Generating Knowledge for reachability test
for combined service of service a and service b



【POTS】 【service a】 【service b】

union

Knowledge

{idle,dialtone},{dialtone,talk},…

Problem 3

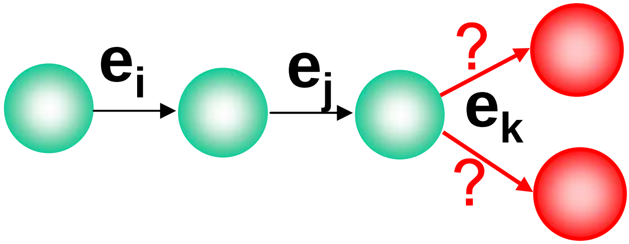
Static Detection Algorithm of Feature Interactions

Classification of Interactions

Classification based on FSM Model

Logical Int. : can be identified by State Transition Diagrams

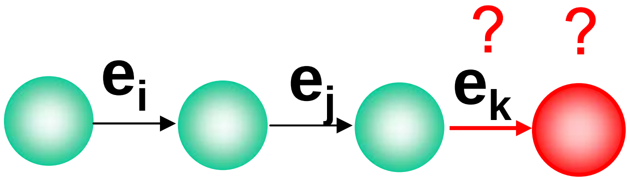
Non-determinacy, Dead lock, Live lock



Non-determinacy

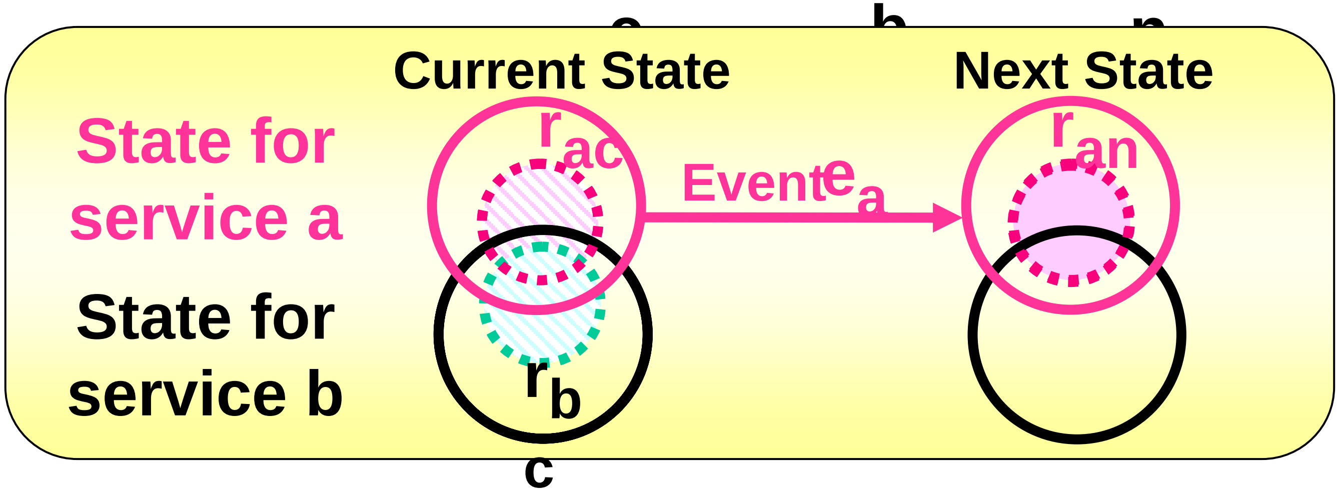
Semantic Int. : can be identified by meaning of State Transiti

Occurrence of abnormal state/transition
Disappearence of normal state/transition



Static Detection Algorithm

	Pre-cond	Event :	Post-cond
rule for service a	r_{ac}	e_a :	r_{an}
rule for service b	r_{bc}	e_b :	r_{bn}



Judging Formula :

$$\{(e_a \neq e_b) \wedge [\{(r_{bc} - r_{ac}) \cup r_{an}\} \not\supseteq (r_{bc} \cup \Delta r_{ac})]$$

$$\vee \{(e_a = e_b) \wedge [\{(r_{bc} - r_{ac}) \cup r_{an} \not\supseteq (r_{bn} \cup \Delta r_{ac})\}$$

$$\vee \{(r_{ac} - r_{an}) \not\supseteq (r_{bc} - r_{bn})\}]$$

→can be judged solely by specifications

Contents

1. Background and Problems
2. Problems and Their Solutions
   - Terminal Assignments
   - Reachability Test
   - Static Detection Algorithm
3. **Implementation and Evaluations**
4. Application to VoIP

Evaluation Items

- **Coverage: As close as possible to 100 %**
- **Redundancy:**
  Detecting what is not actually interaction
- **Detection time:**

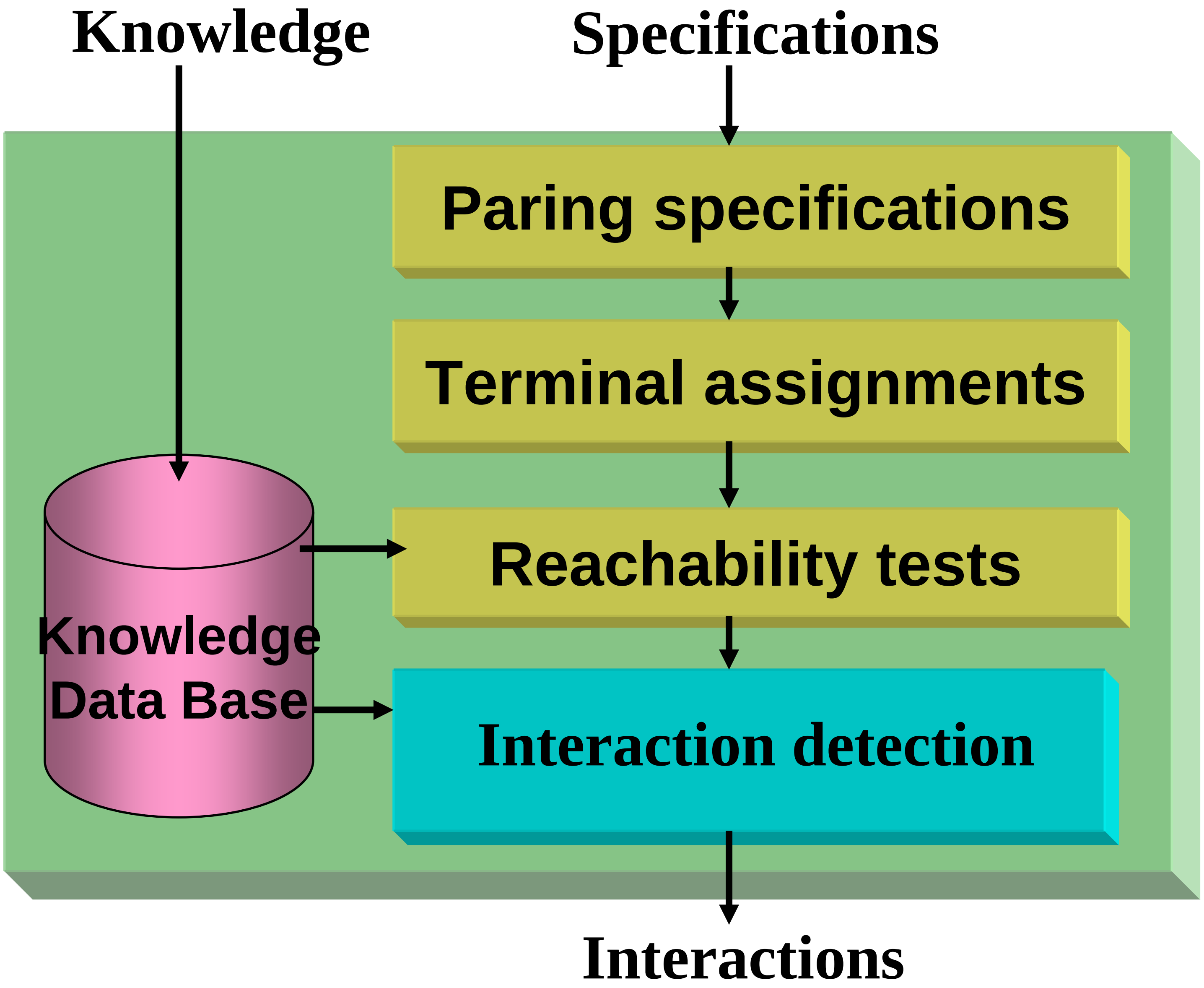
Bench Mark

FIW98 contest results published in 2000

**12 services: CFBL, CND, INFB, INFR,
INTL, TCS, TWC, INCF,
CW, INCC, RC, CELL**

FIW2000 contest results could not used because of lack in detailed information: scenario where interactions occur.

Detection System



Coverage and Redundancy

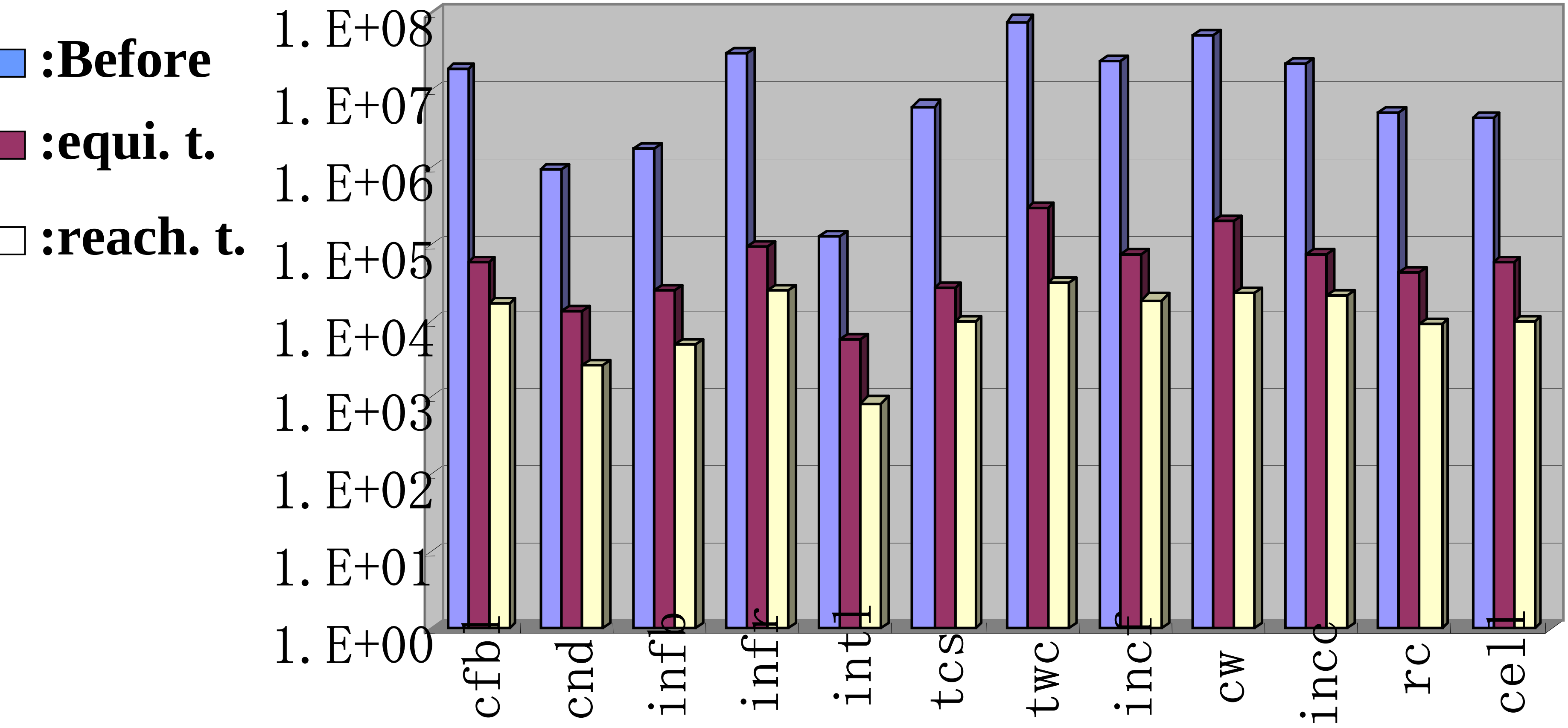
The number of interactions detected: 2,650

Including all interactions described in the bench mark

No redundancies: miss detection,
duplicated detection

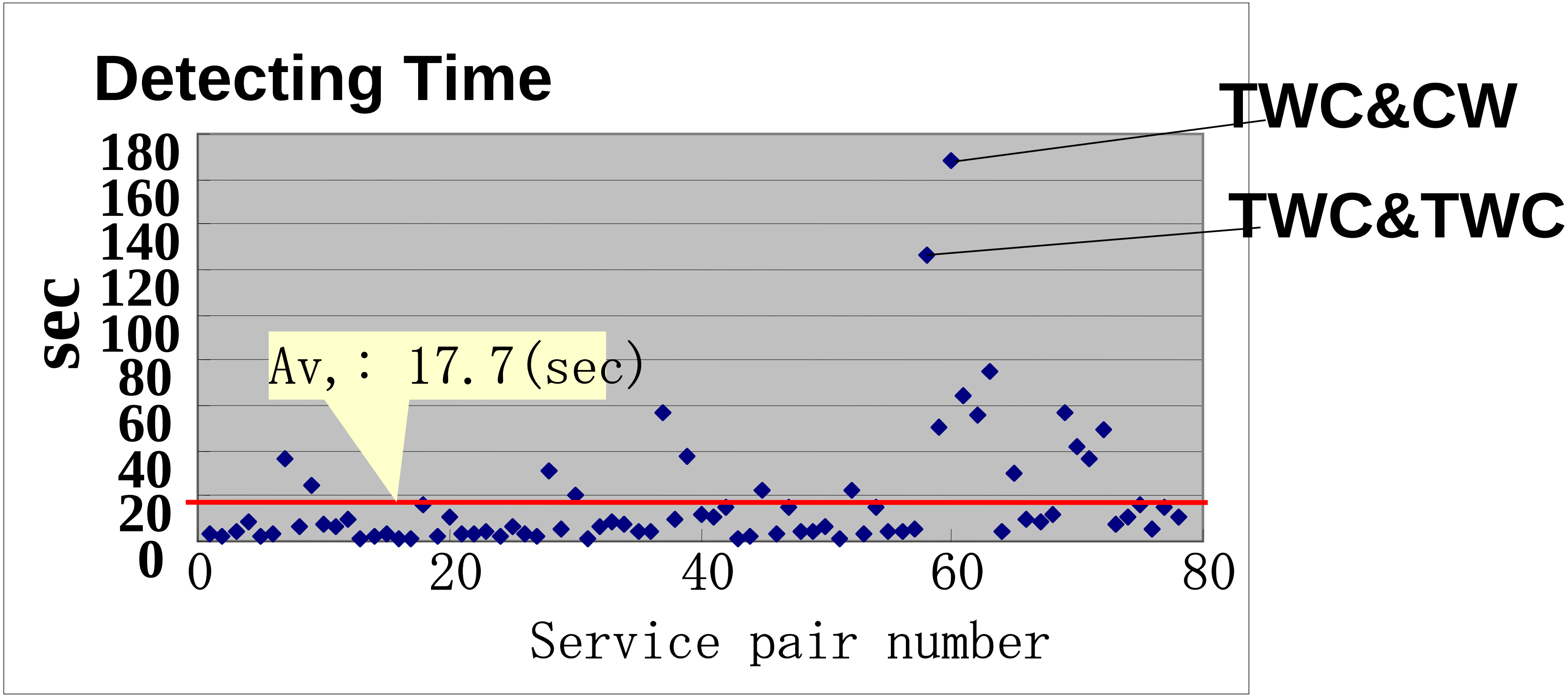
Filtering Effects

The number of testing subjects



reduced to 0.4 % by deleting equiv. term. assignments and reduced to 0.07 % after reachability test.

Detecting Time



- **Mean time for one pair of services: 17.7 sec.**
- **Total time for 12 services: 23 min.**

Evaluations

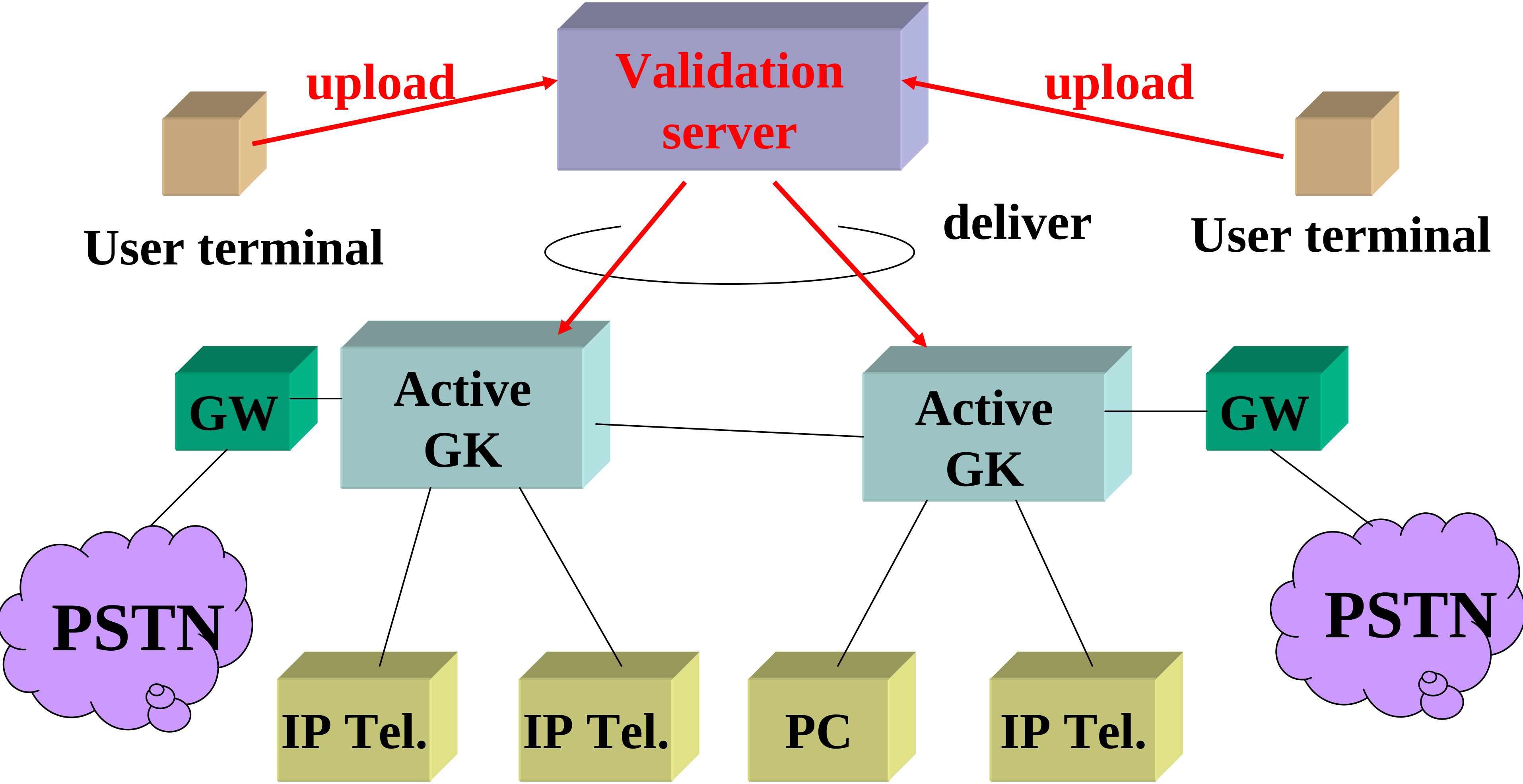
- Coverage: 100% based on the bench mark
- Redundancy: no redundancies
- Detection time: 17.7 sec. ;
  mean time for a pair of services

Effective detection system

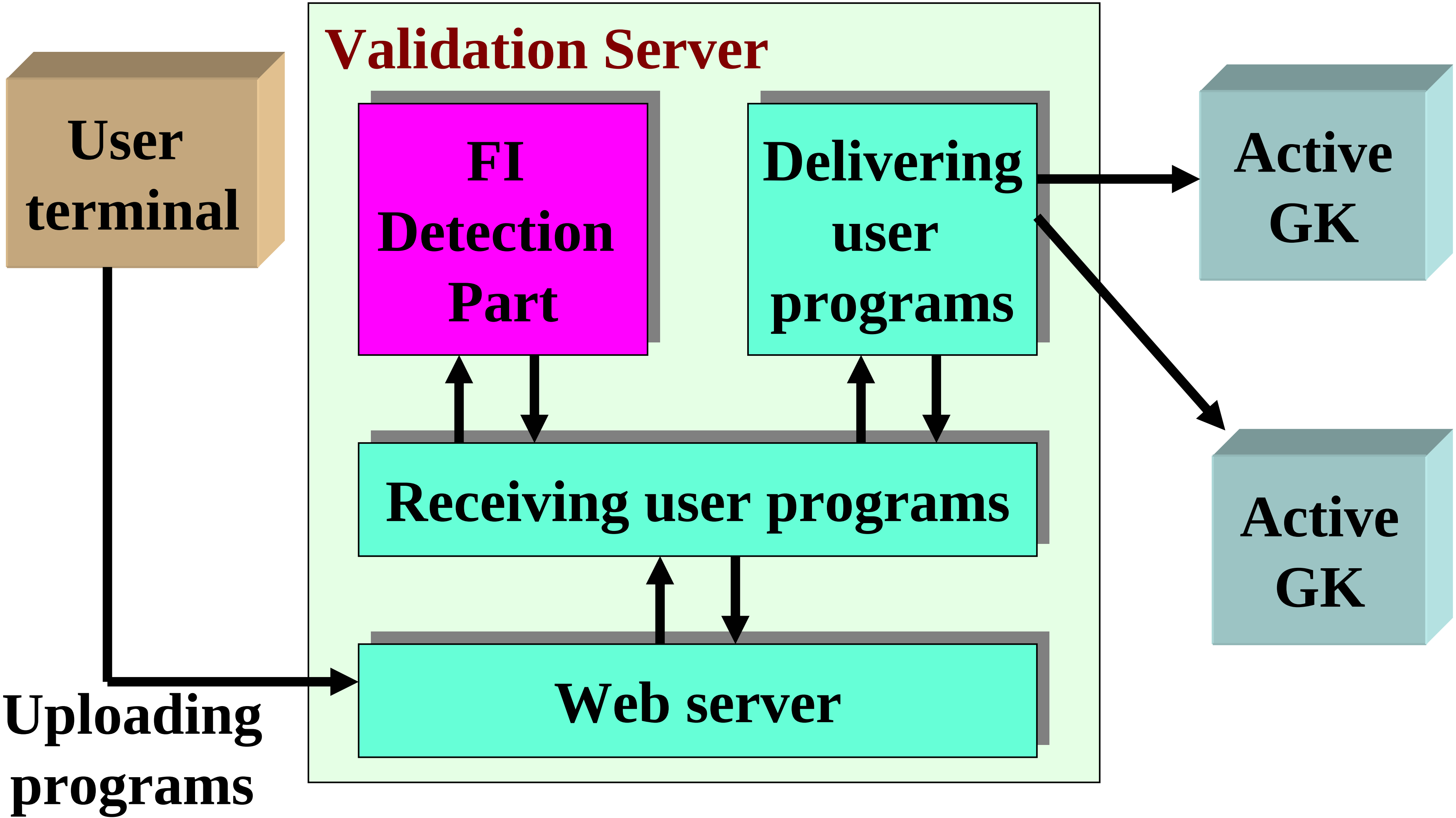
Contents

1. Background and Problems
2. Problems and Their Solutions
   - Terminal Assignments
   - Reachability Test
3. Implementation and Evaluations
4. Application to VoIP

Active Network for VoIP



Experimental System Structure for Validation Server



Future Work

- Interaction resolution algorithm
  - for selecting interactions to be resolved actually
  - for automatic resolution or assisting resolution
- Application to other than telephone services
  - Home network
  - Ado-hoc network
  - Data base system
  - …

Thank you for your kind attentions.

ESTR(2)

Syntax:

Pre-condition event : Post-condition , { action }

Pre-condition: **conditions for state transition**

event: **triger for state transition**

Post-condition: **state after transition**

Action: **procedure accompanied by state transition**

(send a signal, retrieve database, and so on)

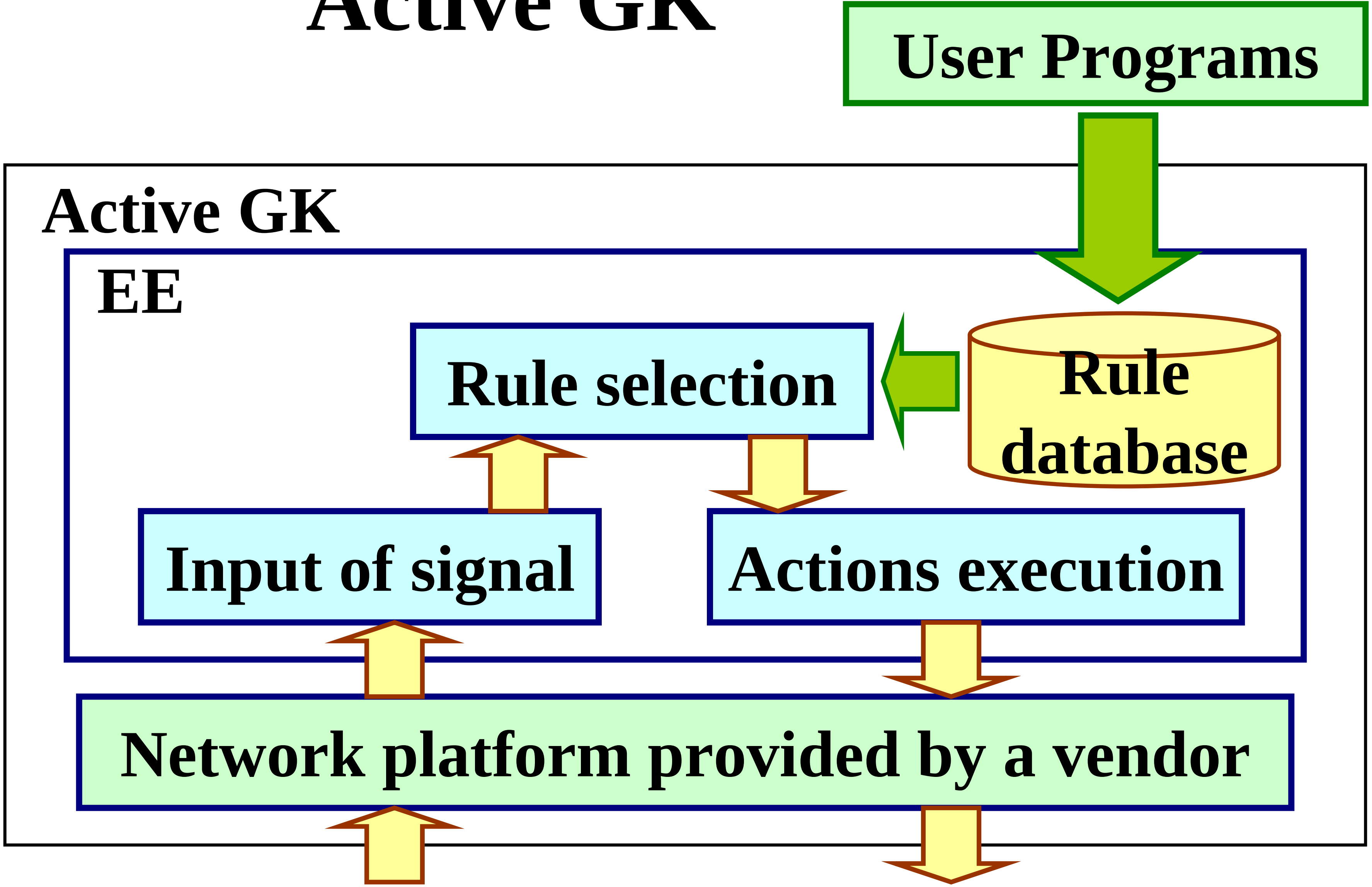
Example;

idle(x) setup(x,y): w-alert(y,x), {Send(setup,x,y)}

Example for ESTR Description

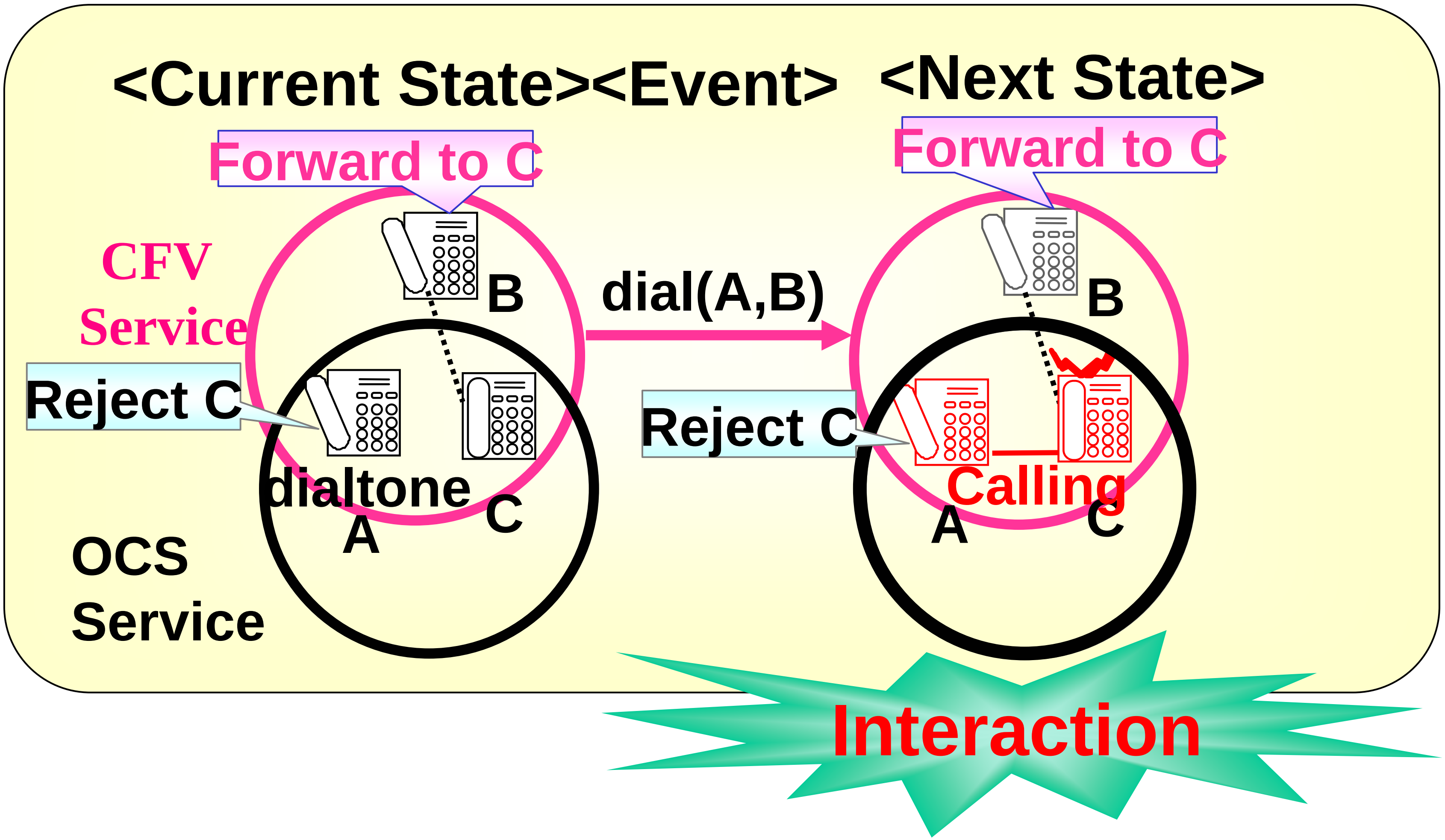
idle(x) arq(x): w-setup(x),{Send(acf,x)}
w-setup(x) setup(x,y): w-arq(y,x),{Send(setup,x,y)}
w-arq(y,x) arq(y,x): w-proc(y,x),{Send(acf,y)}
w-proc(y,x) proc(y,x): w-alert(y,x),{Send(proc,y,x)}
w-alert(y,x) alert(y,x): w-conn(y,x),{Send(alert,y,x)}
w-conn(y,x) conn(y,x): talk(x,y),{Send(conn,y,x)}
talk(x,y) disc(x,y): w-release(y,x),{Send(disc,x,y)}
talk(x,y) disc(y,x): w-release(x,y),{Send(disc,y,x)}
w-release(y,x) release(y,x):
    w-release_conf(x), w-drq(y),{Send(release,y,x)}
w-release_conf(x) release_conf(x): w-drq(x),{}

Structure of Active GK

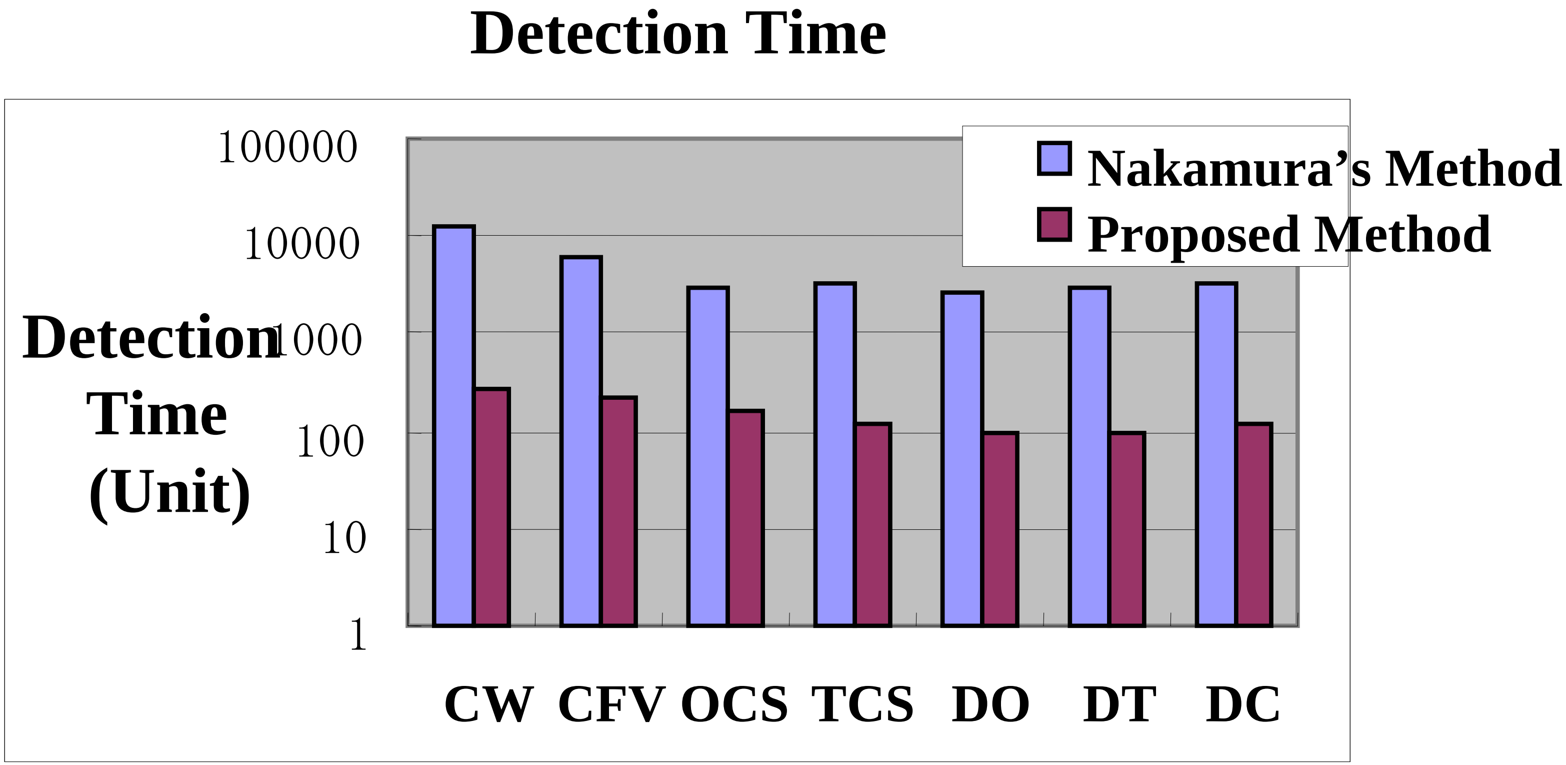


Example for Interaction

Interaction between CFV and OCS



Comparison with Nakamura’s Method



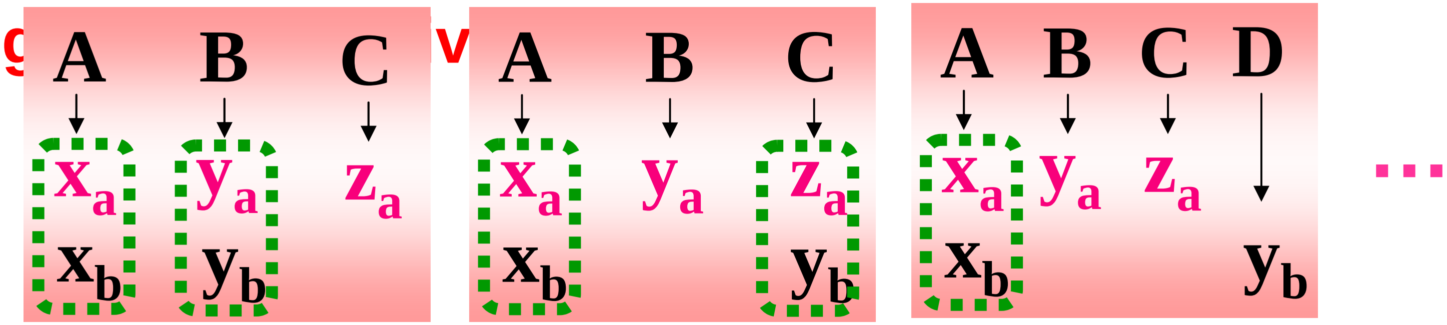
- Can be reduced to 1 60th

DT: reject all terminating call
DO: reject all originating call
DC: direct call (hot line)

Deleting Equivalent Terminal Assignments

Terminal assignments after deleting equivalent ones:

One set of terminal assignment to a combination of terminal variables to which the same terminals are assig v



The number of all terminal assignments $${}_{g}P_{k_a} \times {}_{g}P_{k_b}$$

Deleting equivalent terminal assignments

$$\sum_{t=0}^{k_a} {}_{k_a}P_t \times {}_{k_b}C_t \qquad (k_a \le k_b)$$

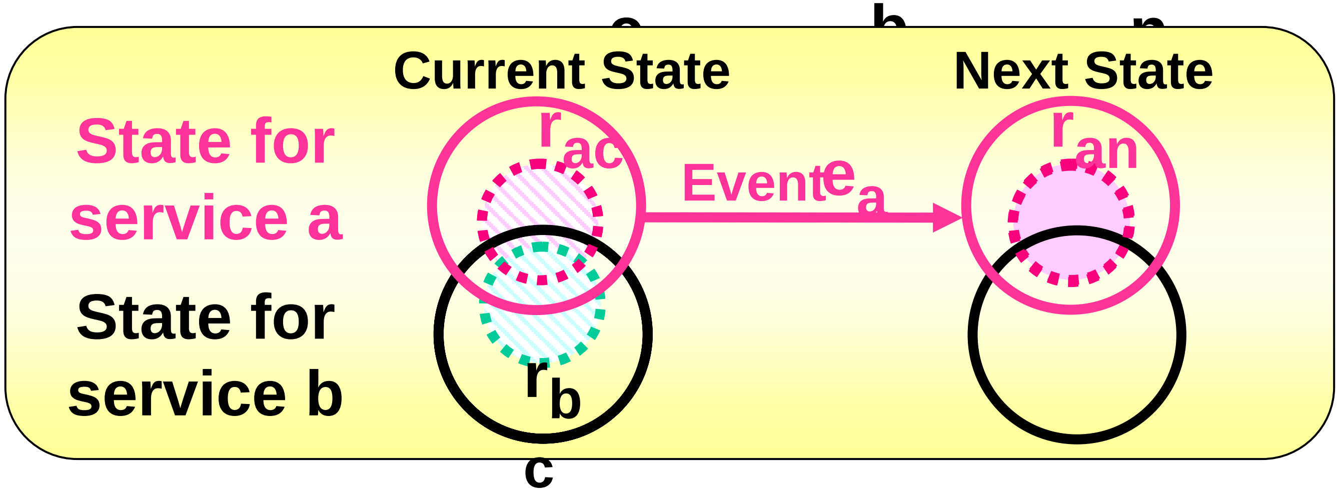
g: The number of real terminals to be assigned

k_a,k_b : The number of terminal variables in service a and b, repectively

t: The number of pairs of terminal variables to be assigned the same term

Static Detection Algorithm

	Pre-cond	Event :	Post-cond
rule for service a	r_{ac}	e_a :	r_{an}
rule for service b	r_{bc}	e_b :	r_{bn}



Judging Formula :

$$\{(e_a \neq e_b) \wedge [\{(r_{ac} \cup r_{bc}) - r_{ac}\} \cup r_{an} \not\supseteq r_{bc} \cup \Delta r_{ac}\}]$$
$$\vee \{(e_a = e_b) \wedge [\{(r_{ac} \cup r_{bc}) - r_{ac}\} \cup \{r_{an} \not\supseteq (r_{bn} \cup \Delta r_{ac})\}$$
$$\vee \{(r_{ac} - r_{an}) \cup (r_{an} \not\supseteq r_{bc} - r_{bn})\}]$$

→can be judged solely by specifications