Control Flow Analysis and Security

Chiara Bodei[1], Pierpaolo Degano[1], Hanne Riis Nielson[2], Flemming Nielson[2]

[1]Dipartimento di Informatica
Università di Pisa
[2]Department of Informatics and Mathematical Modelling
Technical University of Denmark

Language-based technology for security

abbiamo visto che in particolare il pi calculus si presta a modellare protocolli proprio perché ha quella caratteristica dell' operatore di restrizione che ci permette da una parte di creare nuovi nomi, quindi nuovi canali e nuove chiavi se volete e dall'altra di controllarne l' ambito di visibilità, cioè lo scope abbiamo visto quindi come si fa a modellare un protocollo giocattolo tipo quello della rana dalla bocca larga e poi abbiamo visto anche un'evoluzione del pi calculus, proprio orientato alla sicurezza che è lo SPI calculus. Lo SPI calculus semplicemente inserisce nel quadro sintattico anche le primitive di encryption e decryption che se volete potete immaginare un po come l'equivalente di input e output su un canale sicuro e quindi consente di modellare in maniera più vicina i protocolli di sicurezza. A questo punto, quindi, ritorniamo un attimo, sul pi calculus e vediamo una tecnica di analisi statica che può essere utilizzata per indagare proprietà di sicurezza all'interno di contesti in cui la sicurezza è da considerare in particolare l'obiettivo in questo caso sarà capire come fare un'analisi a tempo statico.Il motivo l'abbiamo accennato la volta scorsa, è quello che l'analisi dinamica, cioè con viene fatta a partire dalle evoluzioni dinamiche di un certo sistema, e queste per come le abbiamo viste formalmente nelle ultime lezioni sono rappresentate da un sistema di transizione etichettato e nel momento in cui il sistema accoglie al suo interno in parallelo vari processi e questi hanno molti modi di reagire tra di loro, quindi anche molti interleaving quello che succede è che si ha quella che in gergo viene chiamata l'esplosione degli stati, cioè questo grafo diventa molto ampio e quindi indagare il grafo implica uno sforzo di risorse abbastanza elevato quindi l'analisi statica interviene cambiando il punto di vista e cercando di analizzare quello che succede in previsione, cioè cercando di analizzare la specifica di un sistema, cerca di prevedere quindi approssimando ovviamente quello che può succedere a tempo di esecuzione e quindi esiste una sorta di bilanciamento che uno fa per cui guadagna in efficienza ma perde in precisione. Quindi quello che si fa è di accontentarsi di risposte approssimate che però insomma, se fatte con un certo disegno in qualche maniera sono abbastanza precise, però, per dare risposte significative, le approssimazioni che vedremo sono di tipo conservativo, ovvero si va a fare quella che viene chiamata over approximation o sovrapprossimazione cioè si prevede più di quello che si potrà verificare a tempo di esecuzione. Quindi questa (slide 11) è la natura dell approssimazione a cui stavo facendo cenno cioè se pensate a la parte in azzurro più scuro in basso, nel disegno l'esecuzioni sono un sottoinsieme di ciò che l'analisi mi prevede e mi approssima e questo vuol dire che tutto ciò che l'analisi pensa che possa accadere a run time potrà accadere a runtime, ma non è detto che lo faccia, mentre è escluso che ciò che non fa parte di questo quadro di sovrapprossimazione possa appartenere invece alle esecuzioni dinamiche.

Static Analysis: why?

- There are many questions we can ask about a given program.
- Unfortunately, all interesting questions about the behaviour of a program are undecidable, BUT
- we want to solve practical questions

$\Rightarrow$ approximate answers, still precise enough to fuel our applications.
Approximations are *conservative*: all the errors lean to the same side

abbiamo anche ragionato sul motivo per cui uno ricorre a delle analisi statiche e abbiamo visto che sostanzialmente sono il risultato di una sorta di compromesso, cioè io pago la maggiore economicità dei mezzi che utilizzo in termini di precisione, quindi, mi accontento di risposte approssimate che però lo sono abbastanza per gli scopi che uno vuole ottenere. Approssimazione nel nostro caso di tipo conservativo, ovvero siamo in sovrappassimazione.

Efficiency Concerns

- Transition systems: usually huge $\Rightarrow$ their exploration can be computationally hard.
- Need of obtaining information about the dynamic behaviour, without spending so much.
- There are two alternatives:

Il motivo per cui si spende meno facendo l'analisi statica è che l'analisi dinamica uno dovrebbe farla su tutti i possibili comportamenti, quindi dovrebbe sviluppare il famoso transition system che abbiamo visto nelle ultime elezioni e che soprattutto in presenza di sistemi piuttosto complessi, diventa molto sviluppato e quindi l'esplorazione diventa anche dal punto di vista computazionale molto impegnativa e quindi la possibilità è quella di riguardare tutti i comportamenti prevedendoli rispetto alla lettura della specifica.

... two alternatives:

- Looking through the glass: magic techniques.

- Looking at the system description: static analysis techniques

Static Analysis Techniques

Quindi come tecniche di analisi statica ce ne sono varie, le principali sono l' interpretazione astratta, la control flow analysis e i sistemi di tipo come abbiamo detto la volta scorsa, in tutti e tre i casi, quello che si cerca di ottenere è ottenere informazioni non banale sul comportamento dinamico, senza però pagare il prezzo dell'analisi di tutto le possibile tracce di comportamento e quindi quello che si va cercando sono delle approssimazioni che siano safe e calcolabili del comportamento dinamico e quello che viene fuori è che uno può avere delle proprietà che valgono per ciascuna esecuzione quindi io vado cercando una sovraapprossimazione che comunque mi catturi tutte le possibili evoluzione dinamiche del mio processo, ne cattura così tante che ne cattura anche di non di non effettuabili poi davvero a runtime e l'altro punto di importante di questo tipo di tecniche è che sono a disposizione tutta una serie di metodi e strumenti automatici indecidibili proprio perché , sono tecniche sviluppate da molto tempo, soprattutto nascono in ambito di studio, di compilatori, quindi vengono qui ovviamente reindirizzate sul problema specifico, ma come modalità di approccio sono sviluppate da tanto tempo.

Static techniques { Abstract Interpretation / Control Flow Analysis (CFA) / Type Systems

- Non trivial information about the dynamic behaviour, by simply inspecting the description of the system.
- predict safe and computable approximations of dynamic behaviuor
- analyse properties that hold in every execution
- give a repertoire of automatic and decidable methods and tools

Abbiamo visto che ci sono vari tipi di tecniche statiche e comunque in ogni caso siamo interessati alle predizioni di tipo safe e naturalmente calcolabile e queste sorte di predizioni devono andar bene per tutte le esecuzioni naturalmente si possono ottenere in maniera automatica e decidibile,

In Checking properties

STATICALLY (analyse the TEXT)	DYNAMICALLY (analyse the BEHAVIOUR)
approximate	precise
terminates	may *not* terminate
“low” complexity	“high” complexity
“cheap” tools	“expensive” tools

SOUNDNESS

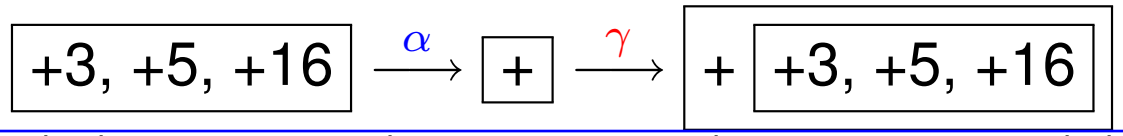
P has the static property implies *P* has the dynamic property

err on the safe side

quindi questo è un po l' interplay tra lo statico e il dinamico vediamo che proprio hanno caratteristiche complementari e laddove si usa un'analisi statica, poi dobbiamo anche avere dei risultati di soundness che ci permettono di estrapolare dalla proprietà vista in termini statici la proprietà corrispondente in termini dinamici

Abstract Interpretation

- It approximates the concrete semantics of dynamic systems, by giving a corresponding abstract semantics.
- It treats the semantic correctness.
- It may use the abstract semantics as a basis for producing automatic tools.



l'interpretazione astratta è un modo di approssimare il comportamento di un sistema e quindi di quella che viene chiamata in questo contesto per contrasto la semantica concreta di un sistema dinamico dandone la corrispondente semantica astratta che cosa vuol dire la semantica astratta? Se mi consentite questa iper semplificazione quello che si va a fare è concentrarsi su alcuni aspetti, dal punto di vista del tipo di proprietà che vogliamo indagare trascurando gli altri e quindi un esempio stupidissimo di interpretazione astratta è quella che mi consente di di stabilire il segno delle operazioni dell interi, indipendentemente dal valore dei numeri coinvolti e questo può essere usato per una serie di motivi per cui, si si parla di funzioni alfa e gamma, per cui la funzione alfa è la funzione che mi astra dai dettagli che non voglio osservare la funzione gamma è quella che mi fa ritornare indietro sugli aspetti più concreti.

Type Systems

invece i type system e li avete visti e sostanzialmente va a creare a stabilire una disciplina di tipo perché in grado di dividere i valori di programmi in tipi e poi va a vedere se ci sono delle violazioni rispetto alla disciplina di tipo che uno si è imposto e qui avete visto information flow, per esempio uno vede che può classificare i dati in tipi di un reticolo, il più semplice, alto e basso e poi vede come vengono gestiti i costrutti del linguaggio e vede se vengono gestite secondo la la caratteristica che uno vuol dare, per esempio nell information flow non voglio che ci sia flusso di informazione dall'alto verso il basso e quindi lo possa andare a controllare con il sistema di tipi. Naturalmente come per l'interpretazione astratta e come anche vedremo la control flow analysis, anche per il type system ho sempre bisogno poi di avere trattata la correttezza semantica, ovvero essere sicuro che quello che io in qualche modo stabilisco a tempo statico, analizzandola con i tipi e la disciplina di tipi, la specifica del mio programma possa essere sicuro che questa si riflette poi sul comportamento dinamico e quindi banalmente, per esempio posso stabilire che se ho un costrutto con l'if dove la condizione non è booleana chiaramente da un errore.

- It divides program values into types and establishes a type discipline.
- It checks whether violations to the type discipline may arise (Type checking). Violations correspond to illegal program behaviour.
- It treats the semantic correctness.

It collects data and, at the same time, it checks them to prove properties.
If 4 : *integer* then **if** 4 **then skip** gives an error.

Control Flow Analysis (CFA)

Properties

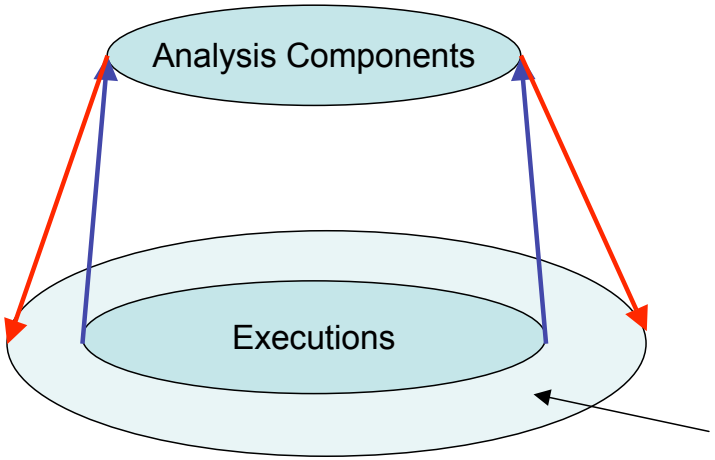
- It predicts approximations (estimates) to the set of values that the objects of a program may assume at run-time.
- It treats (i) semantic correctness, (ii) existence of least estimates and (iii) efficient construction of least estimates.

1. The analysis collects data and afterwards,
2. it checks them in order to prove properties.

A questo punto dobbiamo capire rispetto al pi calculus quali sono i comportamenti astratti che io voglio estrarre dalla mia specifica, dal testo del programma e come sono collegati al comportamento dinamico e quindi come potrò stabilire la mia proprietà di soundness. Se pensate un attimo alle specifiche del pi calculus che abbiamo visto, per esempio per i protocolli, oppure se pensate alla sintassi del pi calculus la mia domanda è, secondo voi quali sono gli ingredienti di una descrizione di un sistema le cui componenti comunicano tra di loro, quali sono i punti che si vogliono in qualche modo seguire, cercando di capire quale possa essere l'esecuzione? Quali sono gli aspetti intanto che io voglio provare a predire? Il pi calculus ha sostanzialmente delle operazioni di input e output che mi mandano avanti tutta l'evoluzione del mio sistema quindi che cosa uno può essere interessato a tracciare? Quello che mi interessa sapere è quand'è che per esempio, due processi comunicano? E che cosa viene comunicato? Quindi io ho bisogno di sapere che cosa fa ciascun processo, quali sono le possibili comunicazioni che possono avvenire? Abbiamo delle comunicazioni che avvengono in due possibili modi: o su canali liberi, cioè su canali il cui nome è libero, quindi il primo processo spedisce sul canale a il valore b e dall'altra parte ci sarà un processo che aspetta sul canale a un qualche valore x e in questo caso cos'è che vorreste predire? Co-a <b>.P | a(x).Q Può succedere che dopo lo scambio, dopo il passaggio su Q associamo il valore x a b, quindi io posso capire due cose: una che sul canale a ci può passare, può essere spedito il valore b e l'altro è che a x può essere legato il valore b. Invece supponiamo di avere questo processo: co-a <b>.P+ co-a<d> | a(x).Q qui cosa possiamo dedurre? Che c'è un'azione non deterministica quindi scambiamo informazione o b o d. Questo da l'idea dell approssimazione, cioè che cosa succede sopra, a vista l'unica azione possibile era quella di sincronizzazione tra l'azione che spedisce b su a e l'azione che lo riceve sui due lati e quindi quello che potevo immaginare è che certamente sul canale a ci potesse passare il valore b e certamente che su x potevo prevedere che potesse finire b, in questo specifico caso, invece, nel secondo quello che succede, come diceva la vostra compagna, è che ho una composizione non deterministica delle azioni per cui ho la possibilità sia di passare b che passare d sullo stesso canale a e ugualmente sul lato di x può essere che accada o l'uno o l'altra cosa quindi che cosa fa l'analisi che sovrapprossima? Suppone che possa essere fatte tutte e due le cose, perché? perché l'analisi deve valere per tutte le possibili esecuzioni e siccome qui sono due, io devo poter prevedere che su a passino sia b che d e ugualmente su x possono essere legate sia b che d. In questo senso si capisce che l'analisi approssimi, perché? Perché nel momento in cui prevede che possono accadere tutte e due le cose in realtà si sta portando dietro due storie che in realtà a run time non ci saranno perché ce ne sarà una sola, quindi se volete ci sarà l'esecuzione in cui viene scelto un ramo e l'esecuzione in cui verrà scelto l'altro. Che cosa succede se io ho la stessa cosa che vi ho scritto sopra nel caso in cui ho questo: co-a <b>.P+ co-a<d> | a(x).co-x<f> |b(w).R |d(z).S qui abbiamo che la continuazione Q è fatta da una coazione su x e poi, in parallelo ci sono altri due processi che fanno rispettivamente un'azione di input sul canale b e l'altro la fa sul canale d. Che cosa potete prevedere dell' evoluzione dinamica di questo sistema? Quindi abbiamo detto che su x può capitare sia il valore b che il valore d questo che cosa vuol dire? cosa succederà a run time nella continuazione del processo? Si crea una sorta di ramificazione perché abbiamo due possibilità quindi o avremo alla fine che il processo si sincronizza su b ottenendo il valore f, e l'altro processo si sincronizza sul canale d ottenendo sempre il valore f facendo la z, ma questo dipende dalla scelta che è stata fatta all'inizio da a. A seconda di quale azione, di quale sincronizzazione c'è stata inizialmente su a si sa se può essere fatta la sincronizzazione su b o su d ma in realtà, siccome entrambe hanno la potenzialità di essere svolte, l'analisi le deve poter prevedere tutte, quindi dovrà poter prevedere un'azione sia su b che su d perché ho bisogno che la mia approssimazione sia valida per tutte le possibili esecuzioni, quindi io non posso più scegliere puntando sul fatto che ne so che è più probabile che venga fatta la comunicazione su d piuttosto che la comunicazione su b. Quindi l'approssimazione nasce dal fatto che dovendo tener conto di tutto quello che può succedere a run time io comincio a ingrossare le fila e quindi sostanzialmente io per poter prevedere tutto ci metto tutto e anche più di quello che succederà perché a run time verrà fatta una sola di queste scelte e non tutte e due e quindi tutte quelle che ho scardinato togliendone una non ci saranno e invece continuano a essere messe tutte insieme perché io ho bisogno di prevedere tutto perché la mia analisi deve essere valida sia nel caso che ci sia un tipo di evoluzione sia nell'altra. Se voi pensate al vostro sistema di transizione, l'analisi che uno fa all'inizio deve poter prevedere, quindi nel grosso tutto quello che succede in ogni punto del grafo, cioè per ogni transazione, quindi il risultato dal punto di vista di ciò che viene passato dai canali, ciò che viene legata alle variabili deve continuare a essere valido in tutto il percorso per tutti i percorsi, cioè sono proprietà che riguardano tutte le possibili evoluzioni. Tornando all'aspetto formale, ora che ne abbiamo dato l' intuizione è quello che succede è che per capire come si evolve un processo di questo un sistema fatto di processi di questo tipo, abbiamo capito che quello che va osservato è quali sono le possibili comunicazioni date dal fatto che appunto, ho un'azione da una parte e la corrispondente coazione dall'altra e quindi una volta che io posso ipotizzare quali sono le comunicazioni da una parte sto dicendo che posso capire che cosa viene passato su un canale e che cosa viene legato sulle variabili, perché qui il problema delle variabili è che io all'inizio qua trovo x e quindi per capire cosa può essere legato a x devo fare tutte le ipotesi su quello che può essere capitato prima dell'azione in cui uso x perché l'azione che uso x è figlia di una precedente, in questo caso contigua, azione di input in cui x viene legato.

The Nature of Approximation

- CFA represents an abstraction of the actual executions
- Concretisation cannot be precise.



Static Event E is included	Dynamic E can happen
Event E is not included	E never happens

From TEXT to ABSTRACT BEHAVIOUR

How is abstract behaviour extracted from program text?

and how is it related to the dynamic behaviour, i.e. soundness?

Programs are annotated (by the compiler) on

- objects (data, vars,...)
  (cf. types: $x : real$; $f : real \times nat \longrightarrow nat$)
- control points (calling points, declaration/use, ...)

e abbiamo visto quindi che la natura dell approssimazione è di tipo sovrappassimato, cioè supponiamo che le approssimazioni quindi siano sostanzialmente più larghe delle esecuzioni reali, cioè i comportamenti se vogliamo essere più precisi, i comportamenti che prevediamo includono quelli che ci saranno a runtime, ma potrebbero anche includere alcune cose che non si verificheranno, quindi l'unica certezza che abbiamo è che se nemmeno nell'allargamento delle approssimazioni è presente un certo comportamento, allora in quel caso sono sicura che a runtime quel comportamento non si verificherà

CFA pattern

- Choose those values of interest for the language.
  - Define the shape of estimates.
  - Define a number of clauses.
- Prove that all estimates are semantically correct.
- Prove that least estimate exist.
- Derive a *constructive* procedure that builds estimates.
- Select a specific dynamic security property and define a static check on estimates.

quindi qual è lo schema di esecuzione dell'analisi per in generale, in particolare per il pi calculus? Io scelgo quali sono i valori di interesse che voglio tracciare, se volete questo mi dà la possibilità di definire la forma di queste stime di queste approssimazioni e a quel punto lì io guidata dalla sintassi posso introdurre una serie di clausole che vedremo che mi aiutano a vedere se le stime che ho ipotizzato sono valide rispetto a quello che voglio ottenere, quindi dovrò dimostrare che queste stime sono corrette dal punto di vista semantico e quindi che la mia approssimazione sostanzialmente riflette quello che succede a run time nel caso specifico della CFA come abbiamo detto, non solo so che queste stime posso dimostrare che esistono, posso anche dimostrare che non ne esiste una migliore di tutte le altre, posso derivare dalle clausole una procedura costruttiva che le vada costruendo e poi, una volta che ho tutto questo impianto stabile, quello che faccio è selezionare una proprietà di sicurezza dinamica e definire qual'è la corrispondente proprietà statica che se vale mi può rassicurare sulla bontà della proprietà dinamica corrispondente.

CFA vs Type Systems

- Type Systems are prescriptive, i.e. they infer types and impose the well-formedness conditions at the same time.
- Control Flow Analysis is mainly descriptive, i.e. it merely infers the information and then leaves it to a separate step to actually impose demands on when programs are well-formed.
- For each property, it is often the case that:
  - a new *ad hoc* type system is necessary, while
  - only a new test on the *same* CFA analysis is needed.

Why (S)pi-calculus?

Quindi, ritornando al PI calculus e vediamo come usare le primitive di comunicazione anche le regole di scope ora qui le vedremo fino a un certo punto e vediamo come analizzare il pi calculus.

Pi-Calculus

- Communication primitives: simple and powerful.
- Scoping Rules: explicitly control the access to channels and to data.

To know the **name** of a channel amounts to having the capability to communicate on it.
The Pi-Calculus does not include cryptographic primitives.

Spi-Calculus

- Primitives for encryption and decryption.
- Directly executable
- Formal semantics.

π-calculus: Remind

Processes:

$P ::= \mathbf{0} \mid \mu.P \mid P \mid P \mid (\nu x)P \mid$ [x=y]P $\mid\ !P \mid P + P$

where: $\mu ::= x(y) \mid \overline{x}y \mid \tau$

ora riprendo un attimo la sintassi in particolare quello che mi interesserà è analizzare le comunicazioni, quindi stabilire che cosa passa dai canali e che cosa viene legato alle variabili

Communication:

$$\frac{P \xrightarrow{x(y)} P', Q \xrightarrow{\overline{x}a} Q'}{P|Q \xrightarrow{\tau} P'\{a/y\}|Q'}$$

ABSTRACT BEHAVIOUR

il pattern della control flow è quello di individuare e tracciare il comportamento di un processo di un sistema attraverso alcune delle sue componenti che osserviamo più da vicino e in particolare abbiamo visto che nel caso del pi calculus il principio di osservazione è quello di osservare le comunicazioni, quindi quello che cercherò di predire quali sono le comunicazioni possibili a run time e quindi avrò in particolare due componenti che mi rappresenteranno questo tipo di informazione e sono la componente RHO e la componente k. La componente k sostanzialmente mi lega in maniera astratta ogni nome a tutti i nomi che possono essere spediti su quel canale sul canale da quel nome, quindi k(a) conterrà la previsione di tutti i vari nomi che potranno essere spediti sul canale a, in maniera complementare, vedrò anche il lato, input attraverso RHO che è la componente che mi dice quali sono i possibili legami a run time di una certa variabile e questo è ovvio che dipenda dai canali e da cosa spediscono i canali.

How is abstract behaviour described? As a pair of abstract domains given as functions

$$(\rho, \kappa)$$

ρ : name $\mapsto \{$set of names$\}$ it can be bound to, and $\rho(n) = \{n\}$ for every free name n.

κ : binder/name $\mapsto \{$set of names$\}$ that can be sent over it.

quindi, come dicevamo, questi sono due i due oggetti che voglio tracciare il mio comportamento dinamico astratto che voglio controllare e questo corrisponde al fatto che nella mia approssimazione avrò due componenti, una viene chiamata RHO e una viene chiamata K sempre in greco e corrispondono alle due cose che voglio osservare. RHO mi dirà sostanzialmente cosa può essere legato alle mie variabili, quindi riprendendo l'esempio di prima io vorrò che RHO di x contenga sia b che d. Invece la seconda componente mi dirà, dato un canale, quali sono i nomi che possono essere spediti su quel canale, quindi, tornando di nuovo alla nostra chat, potremmo dire che ciò che passa da a è sia b che d.

abbiamo visto un esempio molto semplice, abbiamo visto che sostanzialmente RHO mi dice che cosa può essere legato una certa variabile e questo dipende sostanzialmente da cosa può essere spedito sul canale che è quello che fa da soggetto all' azione di input.

ABSTRACT BEHAVIOUR: example

$$P \mid Q \mid R = (\underbrace{\overline{a}d.P' + \overline{a}b.P''}_{P} \mid R \mid \underbrace{a(w).\overline{c}w.Q'}_{Q})$$

This process can evolve as:

- $P' \mid R \mid \overline{c}d.Q'$
- $P'' \mid R \mid \overline{c}b.Q'$

In the first case w becomes d, while in the second one becomes b.
The analysis must predict **both** possibilities:

$$\rho(w) \supseteq \{b, d\}$$

ABSTRACT BEHAVIOUR: example

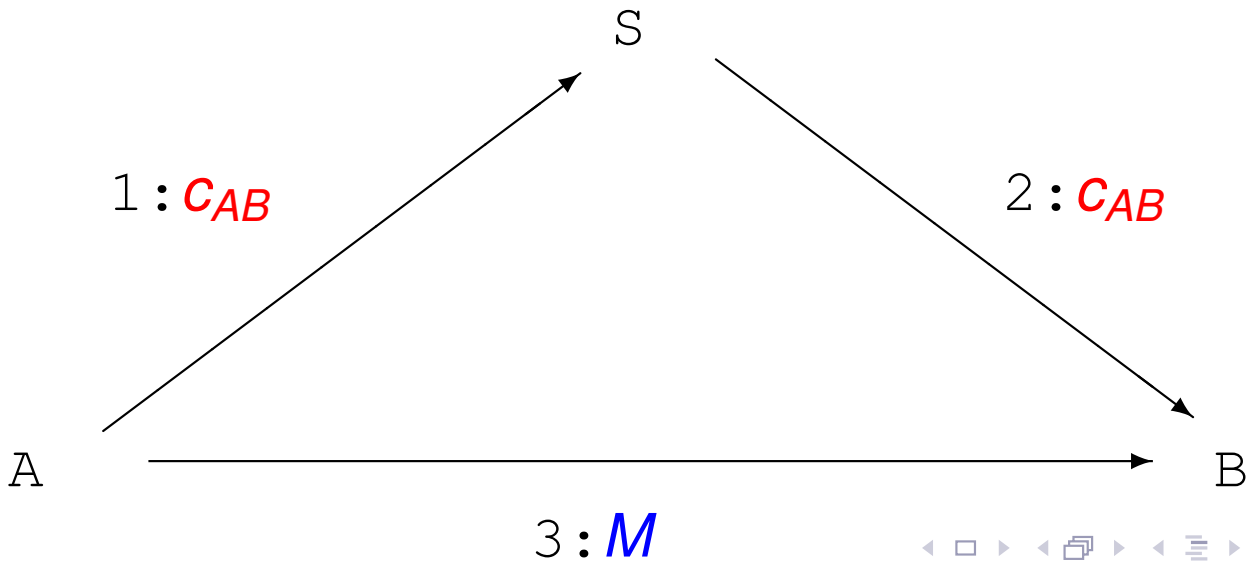
$$P \mid Q \mid R = (\underbrace{\overline{a}d.P' + \overline{a}b.P''}_{P} \mid R \mid \underbrace{a(w).\overline{c}w.Q'}_{Q})$$

	ρ
a	$\{a\}$
b	$\{b\}$
c	$\{c\}$
d	$\{d\}$
w	$\{b, d\}$

	κ
a	$\{b, d\}$
b	$\emptyset$
c	$\{b, d\}$
d	$\emptyset$

CFA Example: Wide Mouthed Frog

$$
\begin{aligned}
P &= (\nu c_{AS})(\nu c_{BS})(\,(A|B)\,|S) \\
A &= (\nu c_{AB})(\overline{c_{AS}}\langle c_{AB}\rangle.\overline{c_{AB}}\langle M\rangle) \\
S &= c_{AS}(x).\overline{c_{BS}}\langle x\rangle \\
B &= c_{BS}(w).w(y)
\end{aligned}
$$



CFA Example: Wide Mouthed Frog (2)

$$
\begin{aligned}
P &= (\nu c_{AS})(\nu c_{BS})(\,(A|B)\,|S) \\
A &= (\nu c_{AB})(\overline{c_{AS}}\langle c_{AB}\rangle.\overline{c_{AB}}\langle M\rangle) \\
S &= c_{AS}(x).\overline{c_{BS}}\langle x\rangle \\
B &= c_{BS}(w).w(y)
\end{aligned}
$$

$$
\begin{array}{llll}
\rho(x) \supseteq \{c_{AB}\} & & \kappa(c_{AS}) \supseteq \{c_{AB}\} \\
\rho(w) \supseteq \{c_{AB}\} & & \kappa(c_{BS}) \supseteq \{c_{AB}\} \\
\rho(y) \supseteq \{M\} & & \kappa(c_{AB}) \supseteq \{M\}
\end{array}
$$

Estimates Validation

When is (ρ, κ) a *valid* estimate? Formally:

$$(\rho, \kappa) \models P$$

When it satisfies a set of logical clauses (one for each syntactic case). Zoom on the principal ones:

$$(\rho, \kappa) \models \overline{x}\langle y \rangle.P \textbf{ iff } (\rho, \kappa) \models P \wedge \boxed{\forall a \in \rho(x) : \rho(y) \subseteq \kappa(a)}$$

The set of names that can be communicated along x includes the names to which y can evaluate ro (i.e. those in $\rho(y)$).

$$(\rho, \kappa) \models x(y).P \text{ iff} (\rho, \kappa) \models P \wedge \boxed{\forall a \in \rho(x), \kappa(a) \subseteq \rho(y)}$$

The set of channels that can be bound to y includes those that can be communicated along x.

Estimates Validation

Sostanzialmente sono regole che mi consentono di cominciare a osservare la mia analisi e controllare se ho previsto giustamente tutte le possibili evoluzioni del sistema poi naturalmente le regole sono guidate dalla sintassi, quindi queste che sono le regole più in particolari, seguono quelle più generali che sono abbastanza intuitive, cioè qualsiasi soluzione io ipotizzi per il processo vuoto vanno sempre bene e poi la seconda la commento tra un attimo, le regole di composizione, cioè p parallelo q, una stima per p parallelo q è valida solo se vale anche singolarmente, sia per p che per q, stesso dicasi per la stima che vale per p + q, cioè per la scelta non deterministica, deve valere singolarmente, quindi in altri termini più formali, si tratta di una soluzione globale, quindi non cambia a seconda del contesto in cui sto vedendo quindi l'analisi è generale e vale per tutti i pezzettini di cui è fatto il mio sistema. In particolare, poi, ulteriore fonte di approssimazione, non vado a vedere, ed è la seconda regola, se un certo numero è ristretto o meno, perché più semplice , così come non vado a considerare il comportamento infinito perché lo assimilo al comportamento del singolo P. Questa è un'ulteriore forma di approssimazione.

$$(\rho, \kappa) \models \mathbf{0} \text{ iff } \mathit{true}$$

$$(\rho, \kappa) \models (\nu x)P \text{ iff } (\rho, \kappa) \models P$$

$$(\rho, \kappa) \models P \mid Q \text{ iff } (\rho, \kappa) \models P \wedge (\rho, \kappa) \models Q$$

$$(\rho, \kappa) \models P + Q \text{ iff } (\rho, \kappa) \models P \wedge (\rho, \kappa) \models Q$$

$$(\rho, \kappa) \models !P \text{ iff } (\rho, \kappa) \models P$$

Note that the analysis does not consider restriction and replication: this is one of the sources of approximation.

Estimates Validation

queste qui le regole che risolvono tutti gli altri casi sintattici ma che sono più semplici soprattutto si può notare che sono soluzioni globali perché la stessa soluzione RHO,k deve valere in tutte le continuazioni e in tutti i possibili momenti di branch che sia branching dovuto alla composizione parallela che alla scelta non deterministica. Abbiamo visto che nonostante si possa ipotizzare una regola anche più semplice per il matching in cui la estimate è valida per i pattern matching se è valida per la sua continuazione, in realtà qui c'è una piccola regola e un piccolo vincolo che rende la nozione più precisa, ovvero l'analisi continua su P solo se c'è almeno una possibilità, almeno potenziale, quindi sempre in termini di sovrapposizione, che x e y assumino a run time almeno una volta lo stesso valore e quindi sia raggiungibile.

$$(\rho, \kappa) \models [x = y]P \text{ iff } \rho(x) \cap \rho(y) \neq \emptyset \Rightarrow (\rho, \kappa) \models P$$

Note that the check proceeds only if the match can be successful, otherwise there is no point in analysing a sub-process that is not **reachable**.

l'altra regola che è interessante è quella del matching, vi ricordate che è il costrutto del pi calcolo che mi consente di arrivare a eseguire il processo p solo nel caso in cui a run time x e y siano legati a due valori uguali. Allora la mia analisi statica che cosa fa? Potrei utilizzare un sistema tipo questo e bypassare il pattern matching perché posso ipotizzare sempre che possa essere fatto nell'ottica della sovrapprossimazione, invece faccio una piccola ottimizzazione qua perché dico vado ad analizzare staticamente il prosieguo del mio processo solo quando so già che c'è almeno una possibilità per x e y a run time di coincidere. Per fare questo in termini statici, banalmente vado a prendere l'intersezione di quello che ho previsto essere legato ad x con quello che ho essere legato a y e analizzo la continuazione solo quando vedo che questa intersezione è diversa dal vuoto.

The Nature of Imprecision

This Control Flow Analysis is Context-Insensitive and is called 0-CFA.

$$P_1 = (a(y)|\ \overline{a}b) \qquad P_2 = (a(y) + \overline{a}b) \qquad P_3 = (a(y).\overline{a}b)$$

- Note that the variable y in P_1 can be bound to b, while in P_2 and in P_3 it cannot.
- Instead, in the CFA estimate, we have that $\rho(y) \supseteq \{b\}$ in all the three cases.

Back to the example

quindi abbiamo visto l'esempio della rana dalla bocca larga, anche in questo caso posso fare la mia approssimazione e anche in questo caso l'esempio è abbastanza semplice vediamo che non solo sto approssimando, ma in questo caso lo faccio anche con una certa precisione, quindi il fatto di barattare la precisione con la semplicità è di nuovo il motivo per cui uno ricorre anche alla zero cfa poi esistono anche analisi contestuali ma sono più complesse da affrontare.

$$
\begin{array}{lcl}
P & = & (\nu c_{AS})(\nu c_{BS})(\ (A|B)\ |S) \\
A & = & (\nu c_{AB})(\overline{c_{AS}}\langle c_{AB}\rangle.\overline{c_{AB}}\langle M\rangle) \\
S & = & c_{AS}(x).\overline{c_{BS}}\langle x\rangle \\
B & = & c_{BS}(w).w(y)
\end{array}
$$

$$
\begin{array}{lcllcl}
\rho(x) & \supseteq & \{c_{AB}\} & \kappa(c_{AS}) & \supseteq & \{c_{AB}\} \\
\rho(w) & \supseteq & \{c_{AB}\} & \kappa(c_{BS}) & \supseteq & \{c_{AB}\} \\
\rho(y) & \supseteq & \{M\} & \kappa(c_{AB}) & \supseteq & \{M\}
\end{array}
$$

- $(\rho,\kappa) \models P \Leftrightarrow (\rho,\kappa) \models A \wedge (\rho,\kappa) \models S \wedge (\rho,\kappa) \models B$
- $(\rho,\kappa) \models A \Leftrightarrow (\rho,\kappa) \models \overline{c_{AS}}\langle c_{AB}\rangle.\overline{c_{AB}}\langle M\rangle \Leftrightarrow$
  $\{c_{AB}\} \subseteq \kappa(c_{AS}) \wedge (\rho,\kappa) \models \overline{c_{AB}}\langle M\rangle \Leftrightarrow$
  $\{c_{AB}\} \subseteq \kappa(c_{AS}) \wedge \{M\} \subseteq \kappa(c_{AB})$
- $(\rho,\kappa) \models S \Leftrightarrow (\rho,\kappa) \models c_{AS}(x).\overline{c_{BS}}\langle x\rangle \Leftrightarrow \{c_{AB}\} \subseteq \rho(x) \wedge (\rho,\kappa) \models \overline{c_{BS}}\langle x\rangle \Leftrightarrow$
  $\{c_{AB}\} \subseteq \rho(x) \wedge \{c_{AB}\} \subseteq \kappa(c_{BS})$
- $(\rho,\kappa) \models B \Leftrightarrow (\rho,\kappa) \models c_{BS}(w).w(y) \Leftrightarrow \{c_{AB}\} \subseteq \rho(w) \wedge (\rho,\kappa) \models w(y) \Leftrightarrow$
  $\{c_{AB}\} \subseteq \rho(w) \wedge \{M\} \subseteq \rho(y)$

Note that $(\rho,\kappa) \models \mathbf{0} \Leftrightarrow \mathit{true}$ and is not reported