Full Abstraction for Expressiveness: Past, Present and Future

Daniele Gorla

"Sapienza", Università di Roma

Bertinoro, June 18th, 2014

Overview

- *Absolute vs Relative Expressiveness (encodings)*
- *Full abstraction: history* PAST
  - *In denotational semantics*
  - *In expressiveness*
- *Full abstraction: myths and facts* Present
  - *False negatives*
  - *False positives*
  - *On the possibility of having a theory of full abstraction results*
- *Conclusions* future

Presentation based on:

- *D.Gorla, U.Nestmann: "Full abstraction for expressiveness: history, myths and facts"*
- *J.Parrow: "General conditions for full abstraction"*

Absolute vs Relative Expressiveness

- *Absolute expressiveness:*

  “*What can/cannot be rendered in L?*”

- *Relative expressiveness:*

  “*Can L be rendered in another language?*”

  “*Can L render another language?*

Through encodings

Absolute Expressiveness: Advantages and disadvantages

+ Gives a clear feeling of what can be implemented and what cannot
+ Can be used for studying relative expressiveness
  - pick up two languages, one solving a problem and one not
  - find encodability criteria that map a solution in the source into a solution in the target
  - claim that there exists no encoding of the source in the target respecting the criteria
– Difficult to use
  - difficult to properly define the problem
  - difficult to find a solution and/or to prove that a solution does not exist
  - difficult to define reasonable encodability criteria and prove that they map a source solution into a target solution
  - the criteria are problem-driven
– Every problem creates a bipartition of the languages (hierarchies of languages call for several separation problems)

Relative Expressiveness

To compare two languages L1 and L2, try to translate one in the other

1. If L1 can be translated into L2 and vice versa, then the two languages have the same expressive power
2. If L1 can be translated into L2 but not vice versa, then L2 is more expressive than L1
3. If L1 cannot be translated into L2 nor vice versa, then L1 and L2 are incomparable

We cannot accept every encoding, otherwise all results are trivial.

Relative Expressiveness: Advantages and disadvantages

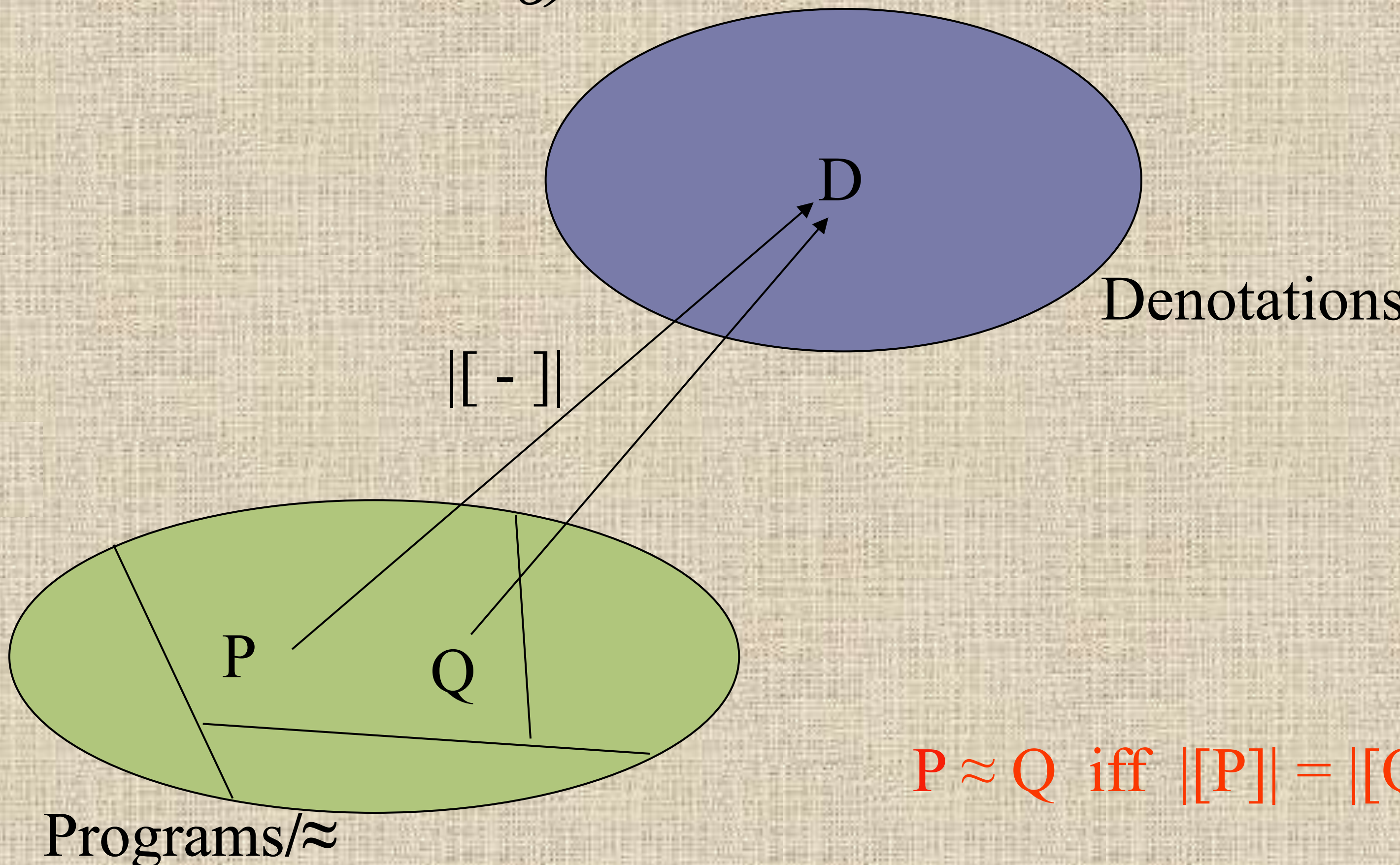
+ Very natural for building hierarchies of languages

+ The encodability criteria are not problem-driven but are 'absolute'

– which criteria define a "good" encoding?

Full Abstraction (in denotational semantics)

Two equivalent programs have the same denotation (i.e., the same meaning)



$P \approx Q$ iff $|[P]| = |[Q]|$

From denotational semantics to expressiveness

- FA relates 2 worlds (programs and denotations) via a mapping

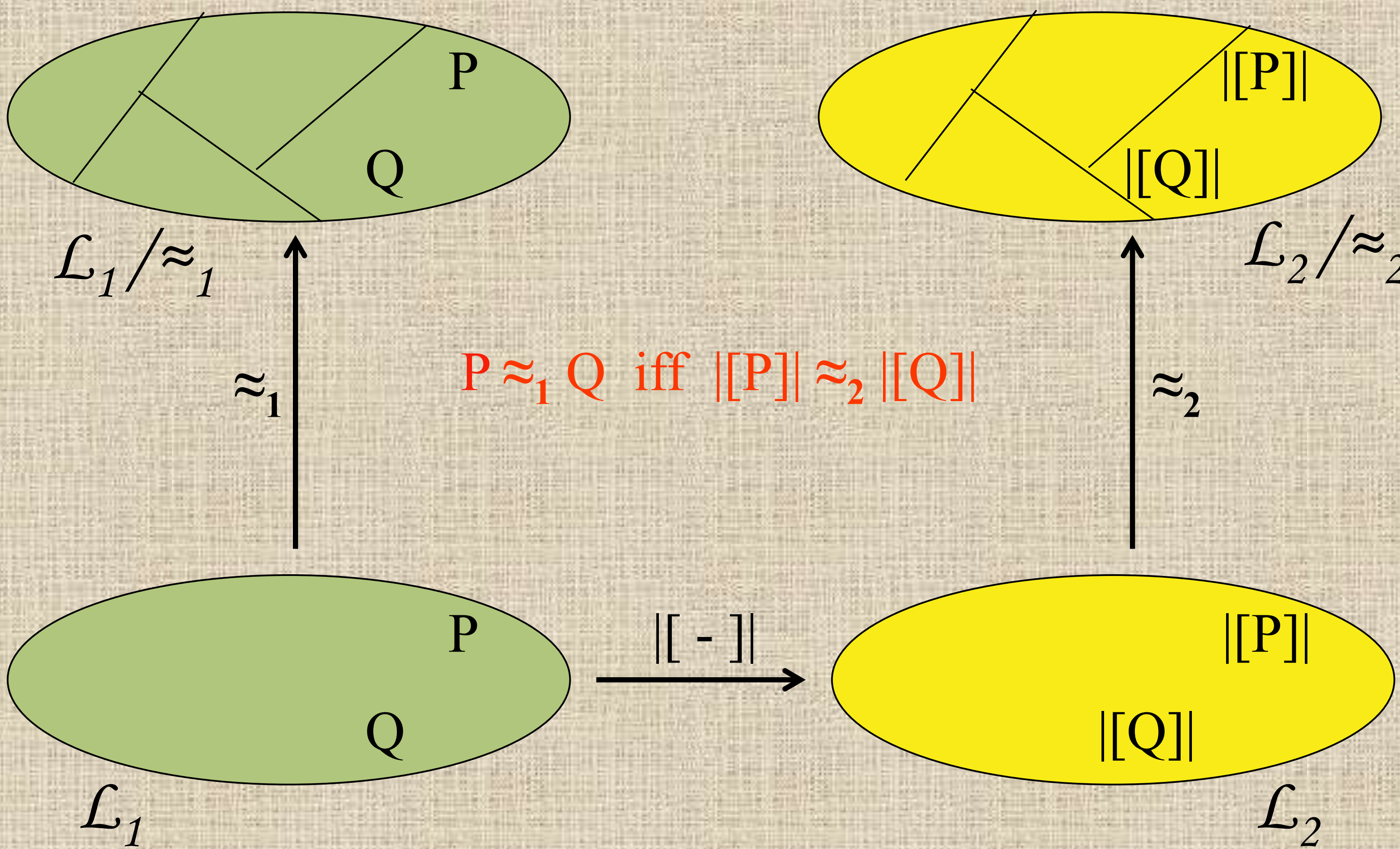
- [Mitchell 1991], [Riecke 1991], [Shapiro 1991] adapted this notion to expressiveness:
  - Mapping = Encoding
  - 2 worlds = 2 different programming formalisms

- In the first setting, one world (denotations) is more abstract than the other (programs)
  - It is possible that different programs have the same denotation

- In the second setting, both worlds are very concrete
  - different programs have different encodings → equivalences on both worlds to abstract away from details

Full Abstraction (in expressiveness)

The encoding respects and reflects the quotient induced by the equivalences in the source and target language



Full Abstraction in Process Calculi

- *Since the early '90s, it has been the reference criterion of several papers on expressiveness for process calculi:*

  - *[Sangiorgi 1993], [Fournet, Gothier 1996], [Victor, Parrow 1996], [Boreale 1998], [Merro 1998], [Amadio 2000], …*

  - "*we assess the relative expressive power of miscellaneous calculi from the existence of fully abstract encodings between them*" *[Fournet, Gothier @ POPL1996]*

Full Abstraction in Expressiveness: Advantages and Disadvantages

It is a property related to the observable behaviour of the languages:

+ the encoding agrees with the observational semantics of the languages
– it gives no hints on what/how the languages compute (i.e., their operational semantics)
– it strongly relies on the behavioural equivalences choosen
– unsuited for proving separation results
– what does it say on the quality of the encoding ??

"Good" Encodings enjoying Full Abstraction

These are (some of) the true positives of our study:

because FA is expected to hold   because FA holds

- [Mitchell 1991]: `let` encodable into untyped λ;
  recursive types into non-recursive ones (always in λ)
- [Riecke 1991]: call-by-name and lazy into call-by-value;
  call-by-value into lazy
- [Nestmann, Pierce 2000]: input-guarded choices into asynchronouns π
- [Merro 2000] and [Merro, Sangiorgi 2004]: expressiveness of $L\pi$
  ($L\pi$ into $L\pi_I$; polyadic $L\pi$ into monadic $L\pi$)
- [Sangiorgi 1993]: $HO\pi$ into π

"Good" Encodings NOT enjoying Full Abstraction

These are the false negatives of our study.

Example:

Pi $\quad P ::= 0 \mid a(x).P \mid a\langle b\rangle.P \mid P|P \mid (\nu a)P \mid !P$

APi $\quad P ::= 0 \mid a(x).P \mid a\langle b\rangle \mid P|P \mid (\nu a)P \mid !P$

- *Trivial encoding of APi into Pi:*

$$|[\, a\langle b\rangle \,]| = a\langle b\rangle.0$$

is not fully abstract w.r.t.

- $\approx$ *(weak bisimilarity for Pi, as defined by [MPW92])*
- $\approx_a$ *(weak asynchr. Bisimilarity for APi, as defined by [ACS98])*

Indeed, $a(x).a\langle b\rangle \approx_a 0,$

whereas $|[a(x).a\langle b\rangle]| = a(x).a\langle b\rangle.0 \not\approx 0 = |[0]|$

"Good" Encodings NOT enjoying Full Abstraction

- *Honda and Tokoro's encoding of Pi into APi*

(the same holds also for Boudol's encoding):

$$|[\ a(x).P\ ]| = (\nu c)(a\langle c\rangle \mid c(x).|[P]|\ )$$
$$|[\ a\langle b\rangle.Q]| = a(y).(y\langle b\rangle \mid |[Q]|\ )$$

is not fully abstract since $a(x).a(x) \approx a(x) \mid a(x)$ *but*

$$\begin{aligned}|[a(x).a(x)]| &= (\nu c)(a\langle c\rangle \mid c(x).|[a(x)]|\ ) \\ &\not\approx (\nu c)(a\langle c\rangle \mid c(x)) \mid (\nu c)(a\langle c\rangle \mid c(x)) \\ &= |[a(x) \mid a(x)]|\end{aligned}$$

Hint: try to close under context $a(z) \mid -$

- *Milner's encoding of polyadic Pi into monadic one:*

$$|[\ a(x,y).P\ ]| = a(z).z(x).z(y).|[P]|$$
$$|[\ a\langle b,c\rangle.Q]| = (\nu d)a\langle d\rangle.d\langle b\rangle.d\langle c\rangle.|[Q]|$$

is not fully abstract since $a\langle b,c\rangle.a\langle b,c\rangle \approx a\langle b,c\rangle \mid a\langle b,c\rangle$ *but*

$$|[a\langle b,c\rangle.a\langle b,c\rangle]| \not\approx |[a\langle b,c\rangle \mid a\langle b,c\rangle]|$$

The reason behind False Negatives

- An encoding is a protocol (to be carried on in the target language)
- There are target contexts that do not respect the protocol imposed by the encoding
- The equivalences used for FA are usually congruences
- FA can be broken by putting the encoding of equivalent source terms in such target contexts

Solution: Weak Full Abstraction ([Parrow 2008])

- FA holds only for equivalences closed under encoded contexts (that, trivially, respect the protocol underlying the encoding) E.g.: [Boreale 1998], [Palamidessi et al. 2006]
- FA holds only for equivalences closed under typed contexts (where the type system implies conformance w.r.t. the protocol) E.g.: [Yoshida 1996], [Quaglia, Walker 2005]

"Bad" encodings that are Fully Abstract (1)

Let's present the false positives.

1. *Consider*
   - $(\Sigma_1, \Sigma_1 \times \Sigma_1)$
   - $(\Sigma_2, \approx_2)$ *with* Σ_2 *non-empty*
   - *the encoding that maps every* $S \in \Sigma_1$ *to the same* $T \in \Sigma_2$

   Then the encoding is fully abstract !!!

2. *Consider*
   - *any encoding* $[\![-]\!]: \Sigma_1 \rightarrow \Sigma_2$
   - $(\Sigma_1, ker([\![-]\!]))$
   - (Σ_2, Id)

   Then the encoding is fully abstract !!!

"Bad" encodings that are Fully Abstract (2)

Turing machines into deterministic finite automata [Beauxis et al. 2008]:

- Enumerate all (minimal) DFA's: $DFA_1, DFA_2, DFA_3, \ldots$
- Group TM's by their equivalence class: $C_1, C_2, C_3, \ldots$
- Encoding: $\forall i \forall TM \in C_i . |[TM]| = DFA_i$
- It is fully abstract w.r.t. language equivalence (their reference equivalences)

Fully Abstraction (almost) for free

[Parrow 2014]:

Thm1: *Given* $(\Sigma_1, \approx_1)$ *and* $(\Sigma_2, \approx_2)$*, there exists* $|[-]|: \Sigma_1 \rightarrow \Sigma_2$ *fully abstract iff the cardinality of* $\Sigma_2/\approx_2$ *is geq than the cardinality of* $\Sigma_1/\approx_1$*.*

Thm2: *Given* $(\Sigma_1, \approx_1)$ *and* $|[-]|: \Sigma_1 \rightarrow \Sigma_2$*, there exists* $\approx_2$ *s.t.* $|[-]|$ *is fully abstract iff* $\forall s,t \in /\Sigma_1 . s \approx_1 t \Rightarrow |[s]| \neq |[t]|$.

Thm3: *Given* $(\Sigma_2, \approx_2)$ *and* $|[-]|: \Sigma_1 \rightarrow \Sigma_2$*, there always exists* $\approx_1$ *s.t.* $|[-]|$ *is fully abstract.*

On changing equivalences
(i.e., can we have a "theory" of FA results?)

Let $|[\ \text{-}\ ]|$ *be a fully abstract encoding of* $(\Sigma_1, \approx_1)$ *into* $(\Sigma_2, \approx_2)$.
For every $\approx'_1 \subset$ *(resp.* $\supset$*)* $\approx_1$, *there exists* $\approx'_2 \subset$ *(resp.* $\supset$*)* $\approx_2$ *such that* $|[\ \text{-}\ ]|$ *is f.a. w.r.t.* $\approx'_1$ *and* $\approx'_2$.

Let $|[\ \text{-}\ ]|$ *be a fully abstract and not surjective encoding of* $(\Sigma_1, \approx_1)$ *into* $(\Sigma_2, \approx_2)$. *There exists* $\approx'_2$ *different from* $\approx_2$ *such that* $|[\ \text{-}\ ]|$ *is f.a. w.r.t.* $\approx_1$ *and* $\approx'_2$.

→ *How can we compare different FA results?*

Full Abstraction in Expressiveness: conclusions

To sum up:

- full abstraction cannot be considered a criterion for assessing an encoding and, hence, to compare the relative expressiveness of languages
- it is an extra value for an encoding:
  - useful if the target language has an efficient proof-technique for its equivalence
  - useful for compositional development of programs (equivalent source processes behave in the same way in any target execution context)

Conclusions

- *we have given evidences against full abstraction as a criterion for expressiveness*

- *this is an a-posteriori justification for some alternative criteria presented in the literature ([Palamidessi 2003], [Gorla 2008, 2010a, 2010b], [Fu, Lu 2010], [vanGlabbeek 2012])*

OPEN PROBLEMS:

- *find the "right" mix of criteria*

- *a new approach to encodability results: show existence of an encoding without exhibiting it*