The chemical reaction: an effective metaphor

La réaction chimique: une métaphore opérante

Daniel Le Métayer

9 November 2012

PLAN

1. Early days: Gamma
2. Exploring the metaphor in various directions
3. The value of metaphors

Gamma in a nutshell

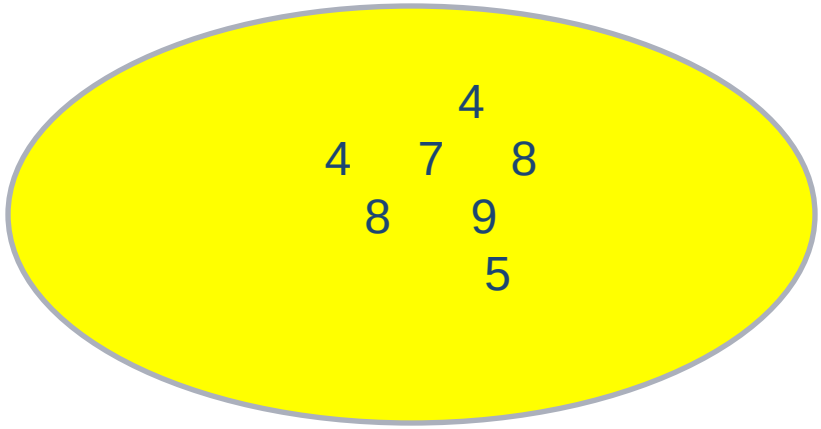
Basic ingredients:

- A solution containing atomic values
- Conditional reaction rules:

  replace $a_1,\ldots, a_n$ by $b_1,\ldots, b_m$ if $C(a_1,\ldots, a_n)$
- Purely local computation steps
- Result: inert solution

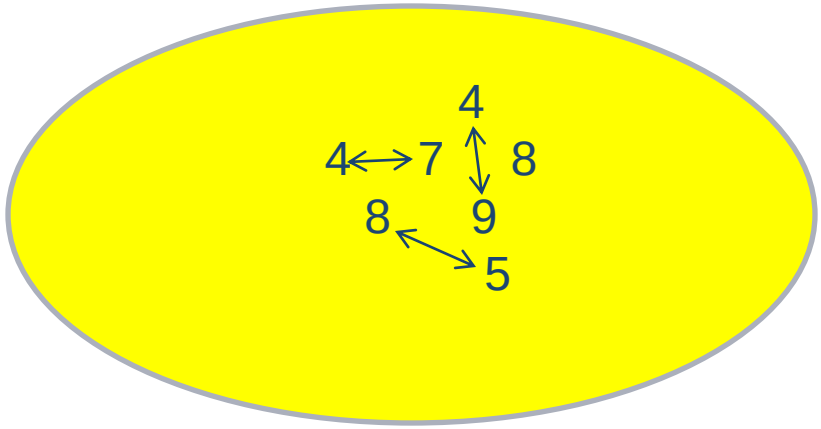
Basic example

Maximum: replace a_1,a_2 by a_1 if $a_1 \geq a_2$



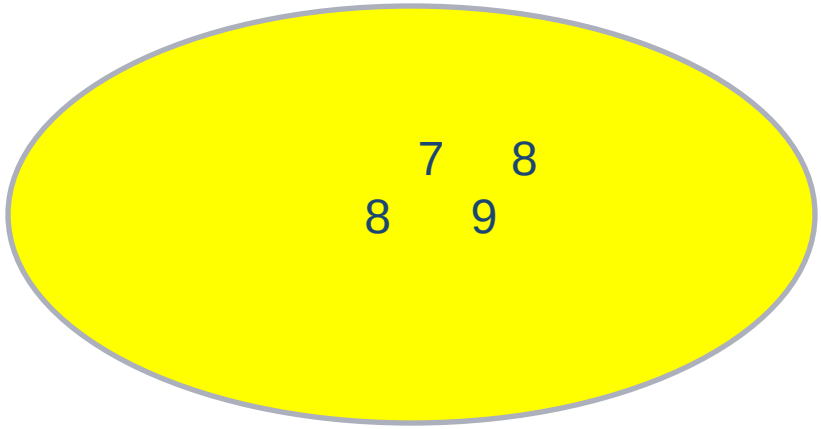
Basic example

Maximum: replace a_1,a_2 by a_1 if $a_1 \geq a_2$



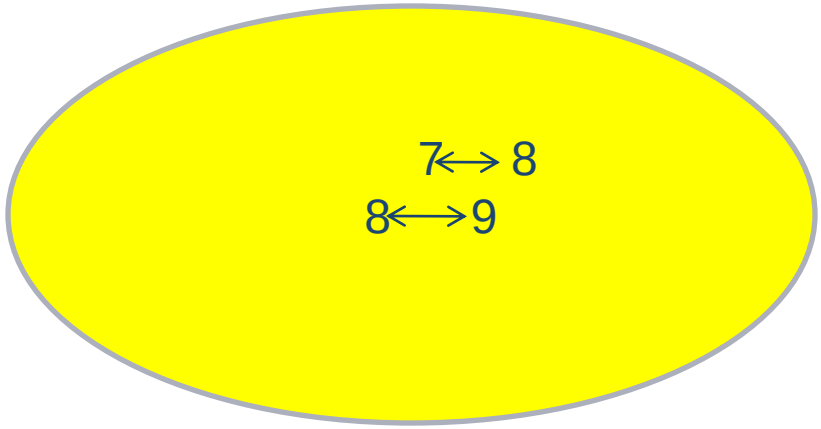
Basic example

Maximum: replace a_1,a_2 by a_1 if $a_1 \geq a_2$



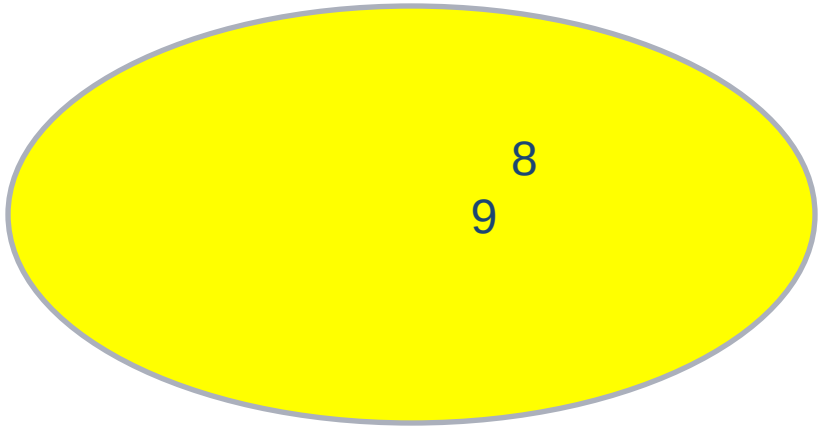
Basic example

Maximum: replace a_1,a_2 by a_1 if $a_1 \geq a_2$



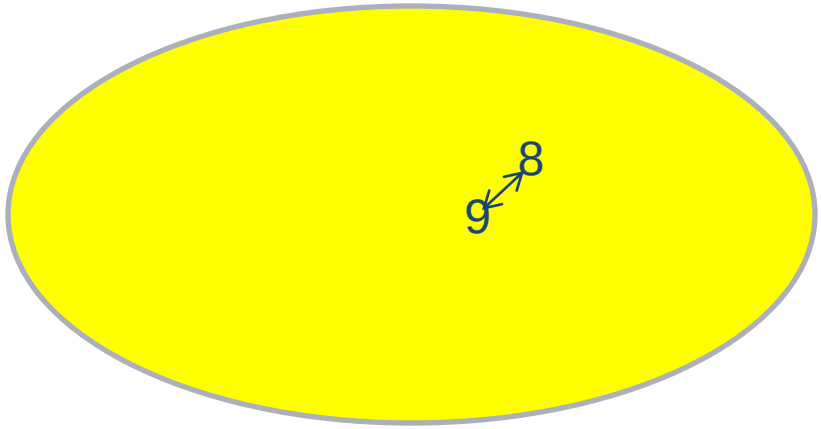
Basic example

Maximum: replace a_1,a_2 by a_1 if $a_1 \geq a_2$



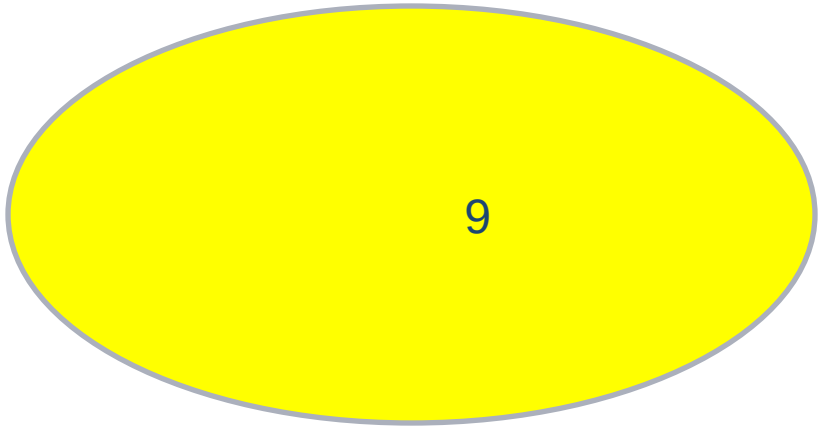
Basic example

Maximum: replace a_1,a_2 by a_1 if $a_1 \geq a_2$



Basic example

Maximum: replace a_1,a_2 by a_1 if $a_1 \geq a_2$



Other basic examples

- Sorting:

  replace $(i_1,a_1), (i_2,a_2)$ by $(i_1,a_2), (i_2,a_1)$ if $i_1 \geq i_2 \wedge a_1 < a_2$

- Eratosthenes' sieve:

  replace (a,b) by (a, a+b/2), (a+1+b/2,b) if b > a+1

  replace (a,b) by a if a = b

  replace a, b by a if div(a,b)

Motivation

Observation: existing programming models lead to overspecify execution control

Detrimental with respect to

- Program verification: need to cope with unrelevant details
- Exploitation of the features of parallel architectures : need to uncover parallelism from a sequential program

Objective: get rid of unnecessary sequentiality

Illustration

Sorting:

replace (i_1,a_1), (i_2,a_2) by (i_1,a_2), (i_2,a_1) if $i_1 \geq i_2 \wedge a_1 < a_2$

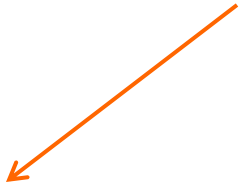
Correctness proof for free:

- Inert solution $\Rightarrow$ elements are well sorted
- Local invariant $\Rightarrow$ the result is a permutation of the initial sequence

Illustration

Sorting:

replace (i_1,a_1), (i_2,a_2) by (i_1,a_2), (i_2,a_1) if $i_1 \geq i_2 \wedge a_1 < a_2$



Correctness proof for free:

- Inert solution $\Rightarrow$ elements are well sorted
- Local invariant $\Rightarrow$ the result is a permutation of the initial sequence

Illustration

Sorting:

replace $(i_1,a_1), (i_2,a_2)$ by $(i_1,a_2), (i_2,a_1)$ if $i_1 \geq i_2 \wedge a_1 < a_2$

Correctness proof for free:

- Inert solution $\Rightarrow$ elements are well sorted
- Local invariant $\Rightarrow$ the result is a permutation of the initial sequence

PLAN

1. Early days: Gamma
2. Exploring the metaphor in various directions
3. The value of metaphors

Exploring the metaphor (1/2)

- Applications (image processing, graph processing, OS kernel, self-organizing systems, service orchestration, etc.)
- Expressing various computation models and languages (Petri nets, Kahn process networks, Linda, etc.)
- Implementations (Maspar, Intel iPSC2, Connexion Machine, PRL-DEC Perle 1, etc.)
- Program derivation and verification (security protocols, shared virtual memory coherence protocols, etc.)

Exploring the metaphor (2/2)

- Richer language (types, combinators, etc.)
- Higher-order versions : chemical abstract machine, higher-order Gamma, the γ-Calculus, HOCL, etc.
- Hybridization with other models and metaphors (Gammalög, λLO, MGS)
- Unexpected directions (software architectures, shape types)

The γ-calculus

$M = x \mid \gamma\,\langle x\rangle.M \mid M_1, M_2 \mid \langle M\rangle$

$(\gamma\,\langle x\rangle.M), \langle N\rangle \rightarrow M\,[N/x]$ if inert(N)

$M_1, M_2 \equiv M_2, M_1$

$(M_1, M_2), M_3 \equiv M_1, (M_2, M_3)$

The γ-calculus

$(\gamma\ \langle x\rangle.\ \gamma\ \langle y\rangle.x)\ ,\ \langle A\rangle, \langle B\rangle$

↓

$(\gamma\ \langle y\rangle.\ A), \langle B\rangle$

↓

A

$\equiv$

$(\gamma\ \langle x\rangle.\ \gamma\ \langle y\rangle.x)\ ,\ \langle B\rangle, \langle A\rangle$

↓

$(\gamma\ \langle y\rangle.\ B), \langle A\rangle$

↓

B

PLAN

1. Early days: Gamma
2. Evolutions, extensions and varied applications
3. The value of metaphors

Some lessons

- The choice of a metaphor can have a dramatic impact on a computational model (reasoning, design, implementation,...)
- Language design: Keep it simple ! (KISS principle, Occam's razor)
- Inherently distributed computational models are still topical (cloud computing, service oriented architectures, multicore architectures, etc.)

Metaphors in computer science

- Countless examples of metaphorical terms: stack, object, package, instruction, pipe, queue, jump, flow, call, message, thread, exception, throw/catch, menu, recycling bin, …
- Criticisms from computer scientists:

  “By means of metaphors and analogies, we try to link the new to the old, the novel to the familiar. Under sufficiently slow and gradual change, it works reasonably well; in the case of a sharp discontinuity, however, the method breaks down.... “

  E. Dijkstra

Metaphors in philosophy of science

Two extreme positions:

Max Black:

- Scientific explanations should rely only on laws, deductive relations, logical generalizations, ...
- Theories should avoid the vagueness and ambiguities of ordinary language, metaphors can hide the truth
- “Thou shalt not commit metaphor!”

Douglas Berggren:

- “Any scientific explanation, though ostensively a logical deduction, can nonetheless rely on a metaphor.”
- Theories need concepts from ordinary language.

Metaphors in computer science

Very different types of use and impact:

- Human Computer Interfaces: desktop, menu, button, recycling bin,…
- Design metaphors: procedure, folder, package, …
- Paradigm shifts: object-oriented computing, data flow languages, chemical reaction, DNA computing, quantum computing, PPSN …

"Because computer science is a science that creates its own subject matter, successful metaphors are used , not to describe the way things are, but the way things should be." (T.R. Colburn, G.M. Shute)

Further metaphors

- A lot of interest in different types of metaphors: chemistry, DNA computing, quantum computing, PPSN …
- Beyond metaphors in « hard sciences » inspiration can also be taken from social sciences
- Specially useful when many new applications are « user-centric » and have a significant social or legal dimension
- Two examples:
  - Control (over data, computation,…): key issue in privacy, for business models
  - Logical causality: key issue for liability, dependable computing

Acknowledgements

Some actors of this story:

Jean-Pierre Banâtre, Gerard Berry, Dominique Bolignano, Gerard Boudol, Paolo Ciancarini, Anne Coutant, Christian Creveuil, Pascal Fradet, Mauro Gaspari, Jean-Louis Giavitto, Chris Hankin, Pieter Hartel, Linpeng Huang, Paola Inverardi, Herbert Kuchen, Huang Linpeng, David Mentré, Hugh McEvoy, Olivier Michel, Gersan Moguérou, Christine Morin, Thierry Priol, Yann Radenac, David Sands, Yongqiang Sun, Marc Vieillot