High-level languages for systems biology

By Michael D. Pedersen. Supervisor: Prof. Gordon Plotkin
LFCS, School of Informatics, University of Edinburgh

The setting - systems biology

1) System: Could e.g. propagate external signals into the nucleus of a cell and activate an appropriate genetic switch (“move”, “mate”, “die”) -- a so called signal transduction pathway.

2) Observations: E.g. connections between chemical species in a pathway or rates of reactions. These are gathered into informal pictorial diagrams by a biologist.

3) Model: A computer scientist creates a formal model corresponding to the pictorial diagram using low-level formalisms such as the Pi calculus, Petri nets etc. Problems: P1 Constructing big models is difficult due to the gap in abstraction level between biology and the low-level formalisms. P2 Pictorial diagrams are ambiguous.

4) Observations: Do they correspond with experimental observations? If no, repeat circle and tweak model. If yes, use model for e.g. medical trials (“does drug X affect part Y of the system?”).

CCS - A Calculus of Chemical Systems

Aim: To design a high-level language with tools for modelling complex biological systems (cf. problem P1), which is intuitive for biologists and has appropriate graphical formal semantics (cf. problem P2).

Key features of language: The basic entities are species with state (e.g. an active/inactive gene). Simple rules are used for defining reactions which consume some species and produce others. Multiple reactions take place in parallel, inside a hierarchy of compartments. General modules can be defined and used repeatedly with different parameter values.

Key features (cont.): Species states are boolean per default, but their data types, and functions on these, can be defined on a per-application basis. Hence information such as DNA sequences (strings), or position in a compartment (integer pairs), can be represented. Semantics is given in terms of coloured Petri nets, a well established mathematical formalism for which analysis and simulation methods are known. Petri nets can be intuitively visualised as a graph of places (species) and transitions (reactions). However, further work is necessary to find the ideal definition of Petri nets for our purposes.

Example - the central dogma of molecular biology

The biology: Production of proteins from information encoded in genes in three steps: T1 Transcription in nucleus. T2 Transport out of nucleus. T3 Translation to protein.

CCS model:

```csharp
module dogma(comp nucl, spec gene, spec mrna, spec prot) {
  nucl[gene + rnap -> gene + rnap + mrna] | nucl[mrna] -> mrna | mrna -> prot
}

comp cell 1E-15 liters;
comp nuc 1E-16 liters inside cell;
spec gene1, gene2, mrna1, mrna2;
spec prot1, prot2;

cell{
  produce two proteins inside cell. dogma(nuc, gene1, mrna1, prot1);
  dogma(nuc, gene2, mrna2, prot2)
}
```

Corresponding Petri nets:

```
Nucleus

T1

T2

T3

gene

map

mrna

prot

Cell

T2

T3

T1

gene1

map

mrna1

prot1

T1

T2

T3

gene2

map

mrna2

prot2
```

Conclusion, status and future work

Conclusion: CCS is a high-level language for systems biology which is intuitive for biologists and has formal semantics in terms of Petri nets.

Status: We currently have a translator implementation in F# from a subset of CCS to a biology XML language (SBML). This allows the use of third-party tools for experimental simulation of models.

Future work: 1) An IDE supporting the development of models in CCS, simulation/analysis of models, and on-the-fly visualisation. 2) Theoretical studies of the Petri net formalism underlying CCS. 3) Application to a nontrivial biological system such as Interferon. 4) A suitable logic for specifying biological properties.