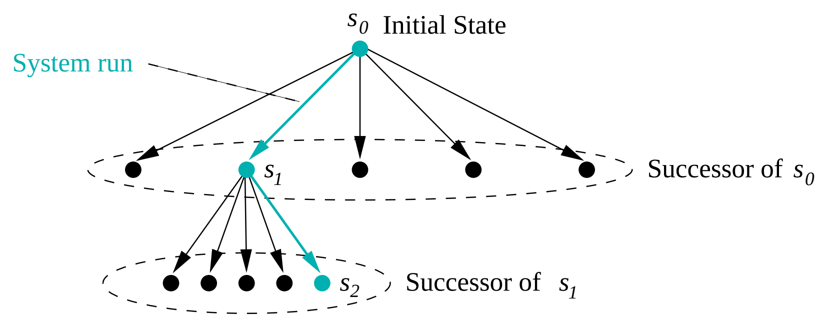
Biological Systems as Concurrent Processes
Part 2: Specifying and Verifying

Wolfgang Schreiner
Wolfgang.Schreiner@risc.uni-linz.ac.at

Research Institute for Symbolic Computation (RISC)
Johannes Kepler University, Linz, Austria
http://www.risc.uni-linz.ac.at

System Behaviors

The possible runs of a system can be described as paths in a tree (actually directed graph) of system states.



From an initial state, multiple runs may emerge (non-determinism).

Computation tree logic (CTL)

A (branching time) variant of temporal logic.

- State formula F:

f	... propositional formula f holds in current state
A P	... P holds in *every* path starting from current state
E P	... P holds in *some* path starting from current state

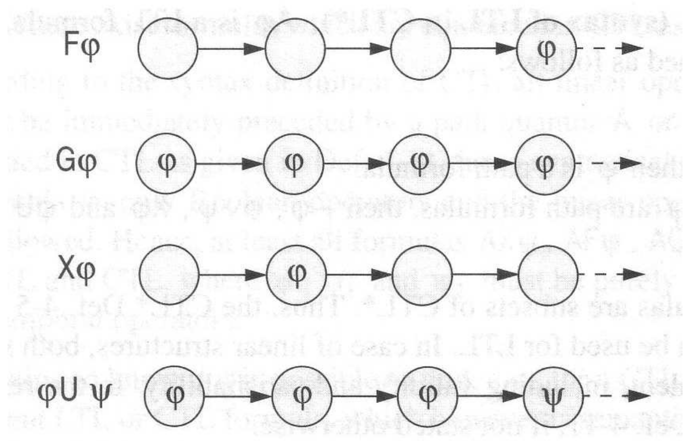
- Path formula P:

F F	... F holds in *some* state of current path
G F	... F holds in *every* state of current path
X F	... F holds in next state of current path
F_1 **U** F_2	... F_1 holds in current path *until* F_2 holds

- Core question: $S \models F$ (does system S satisfy specification F?)
  - Does F hold in every initial state s of S?

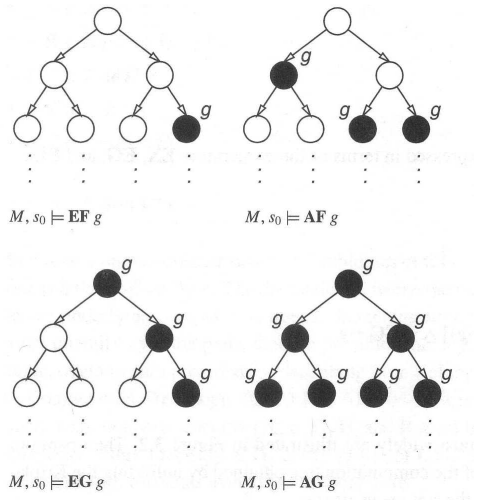
A CTL formula F specifies a system property.

Path Formulas



Thomas Kropf: "Introduction to Formal Hardware Verification", 1999.

State Formulas



g
$M, s_0 \models \mathbf{EF}\ g$
$M, s_0 \models \mathbf{AF}\ g$
$M, s_0 \models \mathbf{EG}\ g$
$M, s_0 \models \mathbf{AG}\ g$

Edmund Clarke et al: "Model Checking", 1999.

Model Checking

System (model) S with a finite number of states.

- A model checker automatically decides the question $S \models F$.
  - Checker needs to investigate only a finite number of states.
  - If property holds, model checker says "yes".
  - If property is violated, model checker produces a counterexample.
    - E.g., for a formula **G** F, a path that violates F.
- Complexity of model checking depends on size of state graph.
  - Complexity is linear in the number of states.
  - State spaces of size 10^{100} (corresponding to 300 bit memory) and beyond can be successfully checked.
- State space explosion
  - In a concurrent system, the size of the state space grows typically exponentially with the number of components.

Model checking is successfully applied to the verification of hardware, communication protocols, and also software.

Systems with Stochastic Behavior

We have already seen that the adequate modeling of biological systems requires some notion of probability.

- System model may be extended by transition rates.
  - Rational value $r > 0$ attached to every edge in state graph.
  - Model becomes a continuous time Markov chain (CTMC).
  - From the transition rates, the probability of various sets of paths originating in a node can be computed.

We want to specify properties of stochastic systems.

Continuous Stochastic Logic (CSL)

Specification logic that includes time and probability.

- State formula F:

f	... propositional formula f holds in current state
$\mathcal{S}_{>c}[F]$	... on the long run, the probability of being in a state in which F holds is greater than c
$\mathcal{P}_{>c}[P]$	... total probability of set of all paths (starting in current state) in which P holds is greater c

  - Relations $>_c, \geq_c, <_c, \leq_c$ with rational number c.
- Path formula P:

$\mathcal{X}^{[t_1,t_2]}\ F$	... F holds in next state of current path and the transition occurs in time $t_1 \leq t \leq t_2$
$F_1\ \mathcal{U}^{[t_1,t_2]}\ F_2$	... F_1 holds in current path until F_2 holds and this will be the case in time $t_1 \leq t \leq t_2$

A specification language for stochastic systems.

CSL Specifications

- *repair* $\Rightarrow \neg$*done*
  - If the current state requires repair, the process is not yet completed.
- $\mathcal{S}_{\geq 0.9}[\mathit{numSensors} \geq \mathit{minSensors}]$
  - From the current state, the long-term probability that an acceptable number of sensors is operational is at least 90%.
- $\mathcal{P}_{\geq 0.9}[\neg \mathit{repair}\ \mathcal{U}^{[0,200]}\ \mathit{done}]$
  - From the current state, with probability 0.9 or more, the process will successfully complete within 200 hours and without requiring repairs.

Interesting kinds of properties can be specified in CSL.

Probabilistic Model Checking

Stochastic finite system model S, CSL formula F.

- A probabilistic model checker decides the question $S \models F$.
  - Based on classical model checking.
  - Extended by computation of probabilities.
    - Equation systems are constructed and solved by iterative methods.
    - Results are exact in that all possible system runs are considered.
- PRISM (Probabilistic Symbolic Model Checker)

  `http://www.cs.bham.ac.uk/~dxp/prism`

  - Automated analysis of quantitative properties of systems:
    - Randomized distributed algorithms.
    - Communication and multimedia protocols.
    - Security protocols.
    - Power management systems.
    - Biological processes.

Technique for verifying (rather than just simulating) biochemical models.

A PRISM Model

```
// nacl.sm: Na + Cl <--> Na+ + Cl-

// ctmc model
stochastic

// number of Na+Cl modules
const int N1 = 10;
const int N2 = N1;

module na
  na : [0..N1] init N1;
  [e1] na>0  ->      na  : (na'=na-1);
  [e2] na<N1 -> (N1-na)  : (na'=na+1);
endmodule

module cl
  cl : [0..N2] init N2;
  [e1] cl>0  ->      cl  : (cl'=cl-1);
  [e2] cl<N2 -> (N2-cl)  : (cl'=cl+1);
endmodule
```

```
// base rates of reactions
const double e1rate = 100;
const double e2rate = 10;

module base_rates
  dummy : bool;
  [e1] true -> e1rate : true;
  [e2] true -> e2rate : true;
endmodule

// rewards: percentage of Na molecules
rewards
  true : 100*na/N1;
endrewards
```

Explanation of the Model

- CTMC model: $\mathrm{Na} + \mathrm{Cl} \leftrightarrow \mathrm{Na}^{+} + \mathrm{Cl}^{-}$.
  - States correspond to number of molecules of each type.
  - Transitions correspond to the reactions between the molecules.
  - Multi-way synchronization of transitions with same name.
    - π-calculus: binary synchronization by channel communication.
- Rate of each reaction:
  - Determined by the reaction's base rate and the concentrations of the participating molecule types.
  - The rate of a synchronous transition is the product of the rates of the local transitions which form the synchronous transition.

    Rate of transition `[e1]`: `e1rate*na*cl`

    Rate of transition `[e2]`: `e2rate*(N1-na)*(N2-cl)`

PRISM employs a state-based modeling language with expressive power similar to the stochastic π-calculus.

Property Specifications

- Basic formulas:

  prop ::= `true` | `false` | `expr` |
  `!`*prop* | *prop* `&` *prop* | *prop* `|` *prop* | *prop* `=>` *prop* |
  `P` *bound* `[` *pathprop* `]` | `S` *bound* `[` *prop* `]` | . . .
  bound ::= `>=`*p* | `>`*p* | `<=`*p* | `<`*p*
  pathprop ::= `X` *prop* | *prop* `U` *prop* | *prop* `U` *time* *prop*
  time ::= `>=`*t* | `<=`*t* | `[`*t*`,`*t*`]`
- Determining the probability of some behavior:

  prop ::= . . . | `P=? [...]` | `S=? [...]`
- Analysis of expected values of rewards:

  prop ::= . . . | `R` *bound* `[` *rewardprop* `]` | `R=? [` *rewardprop* `]`

  - "reachability reward": `F` *prop*
  - "cumulative reward": `C<=`*t*
  - "instantaneous reward": `I=`*t*
  - "steady-state reward": `S`

Analyzing the Model w.r.t. Properties

```
// nacl.csl
const double T;
const int i;
P=? [ true U[T,T] na=i ]
R=? [ I=T ]
R=? [ S ]
```

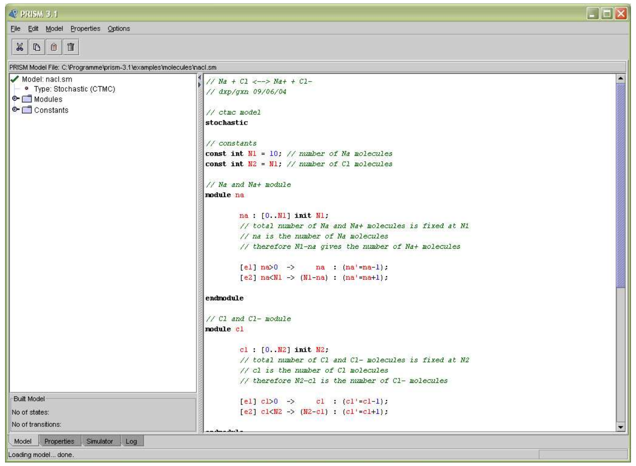
- `P=?[true U[T,T] na=i]`
  - Determine the probability that there are *i* Na molecules at time *T*.
- `R=?[I=T]`
  - Determine the expected percentage of Na molecules at time *T*.
- `R=?[S]`
  - Determine expected long-run percentage of number of Na molecules.

Not only verifying behaviors but determining properties of behaviors.

Using PRISM

- Simulation
  - User-guided/random construction of execution paths.
  - Interactive investigation of states and properties in path.
  - Approximative computation of probabilities/expected rewards.
    - Large number of random paths are generated.
    - Results are evaluated and averaged.
- Verification
  - Model-checking system w.r.t. properties.
  - Determining probabilities/expected rewards of behaviors.
    - Equation systems for these values are constructed.
    - Solutions are numerically computed by iterative methods.
- Experiments
  - Automated execution of multiple model checking instances.
  - Parameterized by values of constants.
  - Generation of graphs depicting probabilities/rewards.

PRISM: Loading a Model

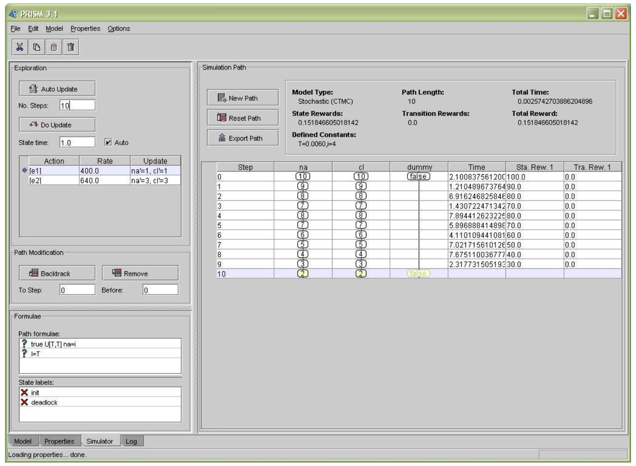


PRISM Model File: C:\Programme\prism-3.1\examples\molecules\nacl.sm

```
// Na + Cl <--> Na+ + Cl-
// dxp/gxn 09/06/04

// ctmc model
stochastic

// constants
const int N1 = 10; // number of Na molecules
const int N2 = N1; // number of Cl molecules

// Na and Na+ module
module na

        na : [0..N1] init N1;
        // total number of Na and Na+ molecules is fixed at N1
        // na is the number of Na molecules
        // therefore N1-na gives the number of Na+ molecules

        [e1] na>0  ->     na  : (na'=na-1);
        [e2] na<N1 -> (N1-na) : (na'=na+1);

endmodule

// Cl and Cl- module
module cl

        cl : [0..N2] init N2;
        // total number of Cl and Cl- molecules is fixed at N2
        // cl is the number of Cl molecules
        // therefore N2-cl is the number of Cl- molecules

        [e1] cl>0  ->     cl  : (cl'=cl-1);
        [e2] cl<N2 -> (N2-cl) : (cl'=cl+1);
```

Loading model... done.

PRISM: Simulating the Model



No. Steps: 10

State time: 1.0 [✓ Auto]

Action	Rate	Update
[e1]	400.0	na'=1, cl'=1
[e2]	640.0	na'=3, cl'=3

To Step: 0 Before: 0

Path formulae:
- true U[T,T] na=i
- I=T

State labels:
- init
- deadlock

Model Type: Stochastic (CTMC) **Path Length:** 10 **Total Time:** 0.0025742703886204896

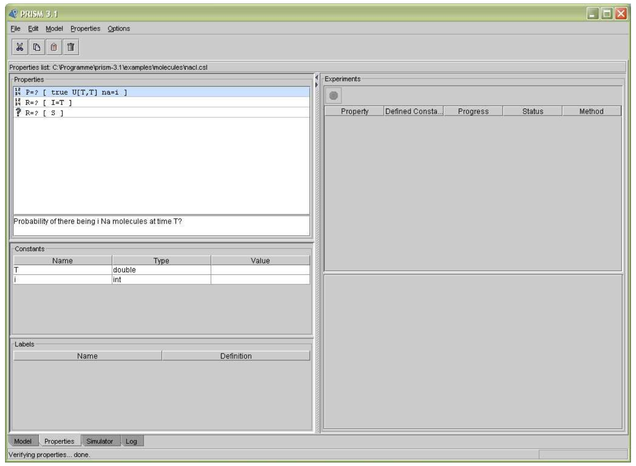
State Rewards: 0.151846605018142 **Transition Rewards:** 0.0 **Total Reward:** 0.151846605018142

Defined Constants: T=0.0060,i=4

Step	na	cl	dummy	Time	Sta. Rew. 1	Tra. Rew. 1
0	10	10	false	2.10083756120(	100.0	0.0
1	9	9		1.210489673764	90.0	0.0
2	8	8		6.91624682584€	80.0	0.0
3	7	7		1.43072247134?	70.0	0.0
4	8	8		7.89441262322£	80.0	0.0
5	7	7		5.89688841489£	70.0	0.0
6	6	6		4.11010944108‍1	60.0	0.0
7	5	5		7.02171561012€	50.0	0.0
8	4	4		7.67511003677‍7	40.0	0.0
9	3	3		2.31773150519:	30.0	0.0
10	2	2	false			

Loading properties... done.

PRISM: Loading Properties



Properties list: C:\Programme\prism-3.1\examples\molecules\nacl.csl

```
P=? [ true U[T,T] na=i ]
R=? [ I=T ]
R=? [ S ]
```

Probability of there being i Na molecules at time T?

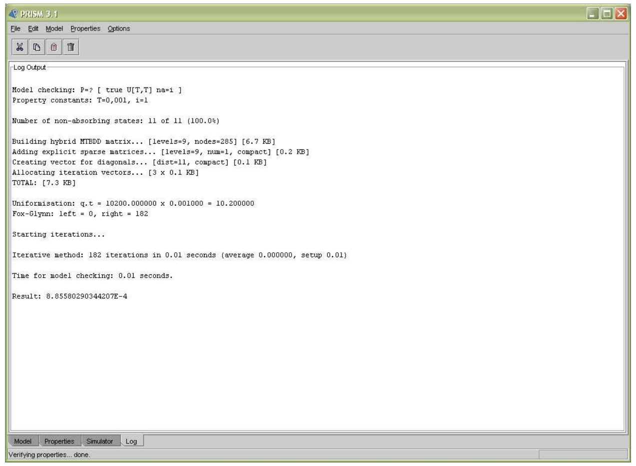
Name	Type	Value
T	double	
i	int	

Name	Definition

Property	Defined Consta	Progress	Status	Method

Verifying properties... done.

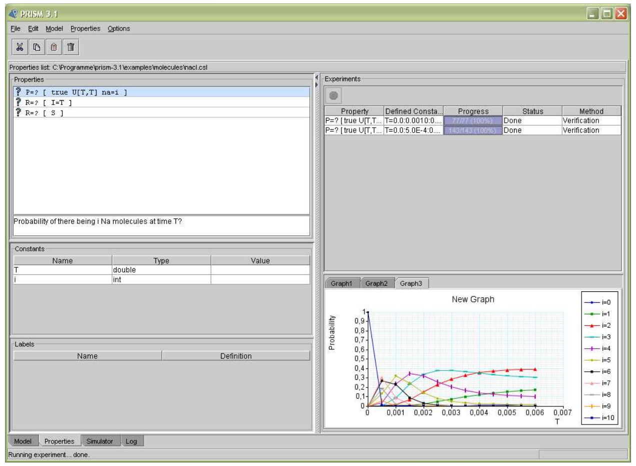
PRISM: Model Checking of Properties



```
Model checking: P=? [ true U[T,T] na=i ]
Property constants: T=0,001, i=1

Number of non-absorbing states: 11 of 11 (100.0%)

Building hybrid MTBDD matrix... [levels=9, nodes=285] [6.7 KB]
Adding explicit sparse matrices... [levels=9, num=1, compact] [0.2 KB]
Creating vector for diagonals... [dist=11, compact] [0.1 KB]
Allocating iteration vectors... [3 x 0.1 KB]
TOTAL: [7.3 KB]

Uniformisation: q.t = 10200.000000 x 0.001000 = 10.200000
Fox-Glynn: left = 0, right = 182

Starting iterations...

Iterative method: 182 iterations in 0.01 seconds (average 0.000000, setup 0.01)

Time for model checking: 0.01 seconds.

Result: 8.85580290344207E-4
```

Verifying properties... done.

PRISM: Running Experiments



Properties list: C:\Programme\prism-3.1\examples\molecules\nacl.csl

```
P=? [ true U[T,T] na=i ]
R=? [ I=T ]
R=? [ S ]
```

Probability of there being i Na molecules at time T?

Name	Type	Value
T	double	
i	int	

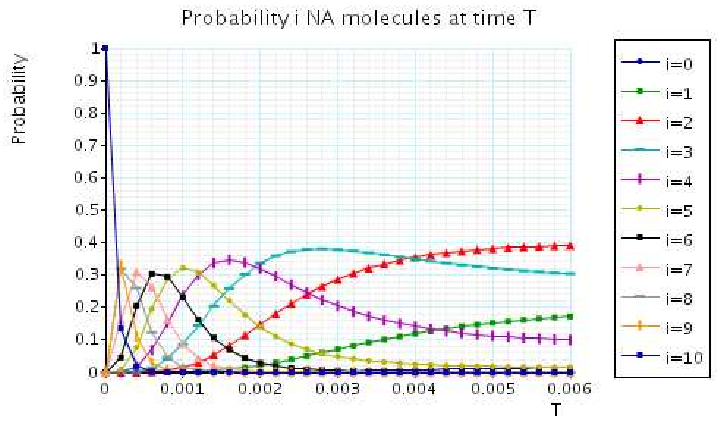
Name	Definition

Property	Defined Consta...	Progress	Status	Method
P=? [true U[T,T...	T=0.0:0.0010:0....	77/77 (100%)	Done	Verification
P=? [true U[T,T...	T=0.0:5.0E-4:0....	143/143 (100%)	Done	Verification

Running experiment... done.

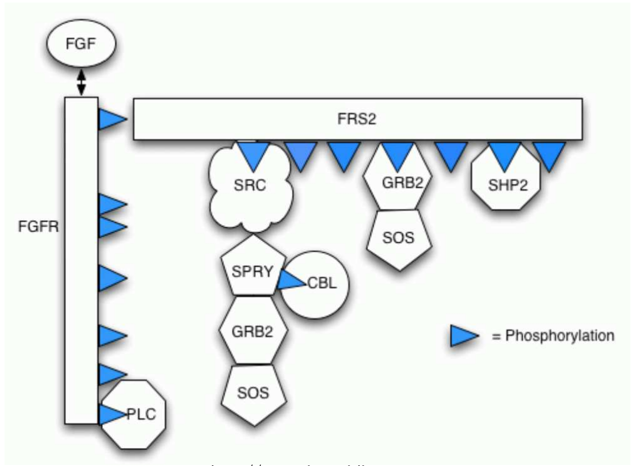
PRISM: Running Experiments

Property: `P=?[true U[T,T] na=i]`



FGF Signalling Pathway

Heath et al: "Probabilistic model checking of complex biological pathways", 2006.



Case Study: FGF Signalling Pathway

- Fibroblast Growth Factor:

  Fibroblast Growth Factors (FGF) are a family of proteins which play a key role in the process of cell signalling in a variety of contexts, for example wound healing. The mechanisms of the FGF signalling pathway are complex and not yet fully understood. In the case study we analyst a model of the pathway based on literature-derived information regarding the early stages of FGF signal propagation and which incorporates several features that have been reported to negatively regulate this propagation.

- Reactions in pathway:
  1. An FGF ligand binds to an FGF receptor (FGFR) creating a complex of FGF and FGFR.
  2. The existence of this FGF:FGFR dimer leads to phosphorylation of FGFR on two residues Y653 and Y654 in the activation loop of the receptor.
  3. The dual Y653/654 form of the receptor leads to phosphorylation of other FGFR receptor residues: Y663, Y583, Y585, Y766.
  4. ...

PRISM Model of FGF Pathway

```
stochastic

formula frs = relocFRS2=0 & degFRS2=0; // frs2 not degraded or relocated
formula fgfr = degFGFR=0; // fgfr not degraded or relocated

module FGF
  FGF : [0..1] init 1; // free

  // fgfr+fgf <-> fgfr:fgf [1]
  [fgf_bind] FGF=1 -> (FGF'=0);
  [fgf_rel]  FGF=0 -> (FGF'=1);
endmodule

module FGFR
  ...
  // fgfr+fgf <-> fgfr:fgf [1]
  [fgf_bind] fgfr & FGFR_FGF=0 -> 5000  : (FGFR_FGF'=1);
  [fgf_rel]  fgfr & FGFR_FGF=1 -> 0.002 : (FGFR_FGF'=0);
  ...
endmodule

...
```

Properties of FGF Pathway

Various model properties specified in CSL and analyzed with PRISM.

- the probability that Grb2 is bound to FRS2 at the time instant T
- the expected number of times that Grb2 binds to FRS2 by time T
- the expected time that Grb2 spends bound to FRS2 within the first T time units
- the long-run probability that Grb2 is bound to FRS2
- the expected number of times Grb2 binds to FRS2 before either degradation or relocation occurs
- the expected time Grb2 spends bound to FRS2 before either degradation or relocation occurs
- the probability that each possible cause of degradation/relocation has occurred by time T
- the probability that each possible cause of degradation/relocation occurs first
- the expected time until either degradation or relocation occurs in the pathway

Summary

Probabilistic model checking is a useful technique for the analysis of biological pathways.

- Pathways are modeled as networks of processes.
  - Properties are described by CSL formulas.
- Quantitative analysis of interactions between pathway components.
  - Calculation of exact quantitative properties for system events occurring over time.
  - All possible behaviors are analyzed.
    - In contrast to methods based on simulation where samples of randomly generated system runs are investigated.

Application of formal verification technique developed in the context of computer science to the domain of biology.