Systematic Literature Review: Approcci di modellazione di sistemi biologici

Mattia Capucci 780653

Alma Mater Studiorum - Università di Bologna
Via dell'Università 50, 47522 Cesena, Italia
mattia.capucci@studio.unibo.it

Sommario La complessità e l'eterogeneità dei sistemi biologici ha portato a uno sviluppo sempre crescente di modelli, tool e framework che potessero descriverne la struttura, i comportamenti e le reazioni. Lo scopo di questa SLR è fornire una descrizione il più sistematica possibile sui principali approcci per la modellazione di sistemi biologici, definendo le caratteristiche di tali metodologie e opportuni esempi.

Keywords: SLR, models, biology, systems biology, networks, boolean networks, pgm, bayesian networks, Petri nets, ODE, PDE, SDE, process algebras, π-calculus, CCS-R, PEPA, Beta-Binders, k-calculus, bio$k$-calculus, markup languages, SBML, PHML, membrane computing, MSR, agents, ELECANS, Repast Simphony, AgentSpeak, Jason

1 Introduzione

I sistemi biologici sono sistemi complessi, spesso non lineari e, di conseguenza, impredicibili. A causa della loro complessità ed eterogeneità, non è possibile definire un modello comune che possa descrivere in modo universale la grande varietà di eventi, meccanismi e reazioni presenti in natura e negli organismi. Nel corso degli anni sono stati proposti diversi modelli, alcuni più generici altri più specifici, che potessero dare una struttura solida a fenomeni biologici e chimici. Data la grande vastità di modelli, tool e framework che sono stati resi disponibili fino a oggi, questa SLR si propone di fornire una descrizione sistematica dei principali approcci di modellazione di sistemi biologici, descrivendoli e fornendo esempi di utilizzo.

La prima parte di questo documento intende definire cosa sia una Systematic Literature Review (SLR) e propone la metodologia utilizzata nel condurla, descrivendola per step successivi.

La seconda parte, invece, si concentra sulla sintesi delle informazioni, elencando i diversi approcci di modellazione identificati nei documenti analizzati e descrivendoli. Per ogni modello sono stati inseriti gli opportuni riferimenti ai testi nei quali esso viene presentato.

2 Systematic Literature Review: definizione e metodologia utilizzata

La Systematic Literature Review (SLR) è un metodo o processo tramite il quale si vuole aggregare e valutare tutta la letteratura rilevante su un determinato argomento e che soddisfi i criteri di eleggibilità specificati per rispondere a una precisa domanda di ricerca. Al fine di ridurre il *bias*, si preparano la logica, le ipotesi e i metodi per il raccoglimento dei dati prima della review e vengono utilizzati come guida per lo svolgimento del processo. Una SLR dovrebbe includere le seguenti caratteristiche:

- Gli obiettivi della ricerca devono essere definiti a priori
- Devono essere esplicitati criteri predefiniti per l'inclusione o l'esclusione della letteratura
- Deve essere definita la strategia di ricerca per la raccolta delle informazioni e per l'esecuzione del processo
- Devono essere definiti i criteri applicati a tutte le fonti e presentati chiaramente nella review
- Deve essere attuata una valutazione sistematica della qualità degli studi inclusi nella review
- Devono essere identificate le fonti della letteratura escluse e fornire una spiegazione della loro esclusione
- Le informazioni devono essere analizzate e sintetizzate
- Fare riferimento a eventuali errori o incoerenze riscontrati nel materiale selezionato

In base alle premesse sopra descritte, la nostra SLR seguirà i seguenti step:

- Definizione degli obiettivi e della *research question*
- Definizione della strategia di ricerca
- Definizione del processo di selezione del materiale
- Valutazione della qualità degli studi
- Sintesi delle informazioni raccolte

2.1 Definizione degli obiettivi e della *research question*

La seguente SLR vuole, attraverso la consultazione tramite motori di ricerca e librerie digitali, sintetizzare tutta la letteratura rilevante che possa rispondere in modo esaustivo alla seguente domanda di ricerca:

"Quali sono i principali approcci utilizzati per la modellazione di sistemi biologici?"

Lo scopo di questa SLR è fornire al lettore una panoramica dei principali approcci utilizzati per modellare sistemi biologici, integrando i diversi studi nei quali essi vengono proposti.

2.2 Definizione della strategia di ricerca

Al fine di condurre una ricerca efficace, sono stati definiti i seguenti punti:

- Fonti da cui trarre il materiale
- Parole chiave da includere nella ricerca
- Limiti applicati alla ricerca
- Screening process

Fonti da cui trarre il materiale. Occorre definire tutte le sorgenti dalle quali si ricavano le informazioni. Nello specifico, nel condurre la seguente SLR sono state effettuate alcune ricerche nei seguenti motori di ricerca o librerie digitali: Google Scholar, Scopus e IEEE Xplore DL. Inoltre, sono stati aggiunti documenti o testi che derivano da conoscenze personali.

Parole chiave da includere nella ricerca. Comprende tutti quei termini che sono stati utilizzati per la ricerca delle informazioni. Le principali keywords, dette *major terms*, utilizzate nella ricerca sono state *model* e *biological system* assieme a tutti i sinonimi o varianti associati e definiti nella seguente tabella.

Major terms	**Sinonimi o varianti**
"model"	"models", "modeling", "modelling"
"biological system"	"biological systems", "systems biology", "biology systems"

In seguito, i termini sono stati inseriti nella seguente query da utilizzare nei diversi portali:

```
("model" OR "models" OR "modeling" OR "modelling")
 AND ("biological system" OR "biological systems"
      OR "systems biology" OR "biology systems")
```

La query è stata, infine, perfezionata per ogni libreria digitale e motore di ricerca.

Limiti applicati alla ricerca. Si indicano tutte quelle condizioni che escludono a priori documenti dalla ricerca o, viceversa, le includono. Il dettaglio dei criteri di inclusione ed esclusione è visibile nel paragrafo successivo.

Screening process. Si specifica come avviene il processo tramite il quale si decide se un documento rispetta i vincoli e i requisiti da noi specificati durante la ricerca. Per la seguente SLR il processo di screening è avvenuto tramite l'analisi del titolo e dell'abstract di ciascun documento.

2.3 Definizione del processo di selezione del materiale

Nel condurre una SLR è importante definire in modo chiaro i criteri di inclusione, ovvero parametri che permettono di includere i documenti nella nostra ricerca e i criteri di esclusione che, viceversa, li escludono.

Criteri di inclusione:

- Fornisce una descrizione dettagliata su uno o più modelli
- Fornisce esempi di applicazione di un determinato modello
- Fornisce una descrizione di un framework, che si basa su un determinato modello

Criteri di esclusione:

- Documenti antecedenti all'anno 2000
- Documenti senza accesso al testo completo
- Documenti scritti in lingua diversa dall'inglese
- Documenti duplicati (ovvero documenti già inclusi, identificati da una precedente ricerca su un determinato portale)

Alcuni criteri di esclusione, come il periodo di pubblicazione e il libero accesso al documento, sono stati possibili a livello di libreria digitale o motore di ricerca, specificando il relativo filtro. Questo ha permesso una prima scrematura dei risultati.

In seguito, il processo di screening è avvenuto analizzando il titolo e l'abstract, verificando che questi fossero coerenti con quanto ricercato.

La prima fase di scrematura ha incluso nella ricerca 35 documenti, che saranno sottoposti ora a una valutazione qualitativa.

2.4 Valutazione della qualità degli studi

I documenti risultanti dal processo di screening sono ora sottoposti a una valutazione di qualità. In questa fase si valutano i diversi testi, analizzandoli nel dettaglio e verificando che essi rispettino concretamente i criteri di inclusione definiti e che, di conseguenza, possano rispondere o contribuiscano a rispondere alla domanda di ricerca prefissata. La fase di verifica della qualità è avvenuta nel seguente modo:

- Scanning iniziale del documento, con analisi delle sezioni

- Ricerca di keywords o frasi all'interno del documento che potessero essere rilevanti per includere definitivamente il testo nella ricerca

- Qualora i punti precedenti non abbiano prodotto un responso concreto, si procede a una lettura veloce del documento per verificare l'effettiva attinenza dello stesso ai criteri di inclusione

La tabella che segue mostra tutti i documenti sottoposti a valutazione. Per ognuno di essi si specifica se viene incluso nella ricerca o meno e il motivo dell'inclusione o esclusione.

Rif. bibliografico	**Incluso nella ricerca**	**Motivazione**
[1]	Sì	Definisce in modo dettagliato diversi modelli di sistemi biologici, con esempi
[2]	Sì	Raggruppa e classifica diversi modelli di sistemi biologici e li descrive, con esempi
[3]	Sì	Definisce modelli nei percorsi di trasduzione del segnale
[4]	Sì	Descrive un approccio ad agenti per la modellazione di sistemi biologici complessi, fornendo un esempio
[5]	Sì	Raggruppa e classifica diversi modelli di sistemi biologici e li descrive, con esempi
[6]	No	Propone algoritmi per simulazioni stocastiche di sistemi biochimici ma non approcci di modellazione
[7]	Sì	Propone le reti booleane come approccio per modellare sistemi biologici
[8]	Sì	Propone diversi approcci per modellare sistemi biologici e li descrive
[9]	No	Troppo specifico nell'esperimento analizzato e poco dettagliato sul possibile modello a cui si ispira
[10]	Sì	Definisce alcune tecniche di modellazione dinamica
[11]	No	Non definisce un modello ma solo una descrizione della Systems Biology
[12]	No	Non definisce concretamente modelli di sistemi biologici ma solo la stima dei parametri
[13]	No	Vengono citati solo alcuni modelli, la cui descrizione è troppo generica
[14]	Sì	Definisce le ODE e i PGM come possibili modelli per studiare le dinamiche del cancro
[15]	Sì	Propone svariati modelli computazionali e tool per sistemi biologici, analizzandoli
[16]	Sì	Propone le reti bayesiane come approccio per la modellazione di sistemi biologici, descrivendole nel dettaglio
[17]	Sì	Propone le reti di Petri come approccio di modellazione di sistemi biologici, descrivendole e fornendo esempi d'uso

Rif. bibliografico	**Incluso nella ricerca**	**Motivazione**
[18]	Sì	Propone le equazioni differenziali ordinarie (ODE), le equazioni differenziali alle derivate parziali (PDE) e le equazioni differenziali stocastiche (SDE) per la modellazione di sistemi biologici, descrivendole
[19]	Sì	Propone modelli stocastici per definire i sistemi biologici, descrivendoli
[20]	Sì	Descrive il linguaggio di markup SBML per la rappresentazione di reazioni chimiche
[21]	Sì	Descrive il linguaggio di markup PHML per descrivere modelli biofisici multilivello
[22]	Sì	Propone il *Membrane Computing* come modello per definire sistemi biologici, descrivendolo nel dettaglio
[23]	Sì	Propone la piattaforma ELECANS per la modellazione di sistemi biologici relativi al cancro
[24]	Sì	Descrive modelli multi-livello e ibridi per sistemi biologici
[25]	Sì	Propone un'interfaccia per costruire modelli basati su process algebras
[26]	Sì	Propone *Hype* come process algebra per descrivere sistemi biologici ibridi
[27]	No	Definisce la stima di parametri ma non un modello specifico
[28]	Sì	Definisce modelli matematici e computazionali di sistemi biologici
[29]	Sì	Definisce MSR come modello per i sistemi biologici
[30]	Sì	Definisce alcune process algebras per definire sistemi biologici
[31]	Sì	Viene proposto bio$k$-calculus per descrivere proteine e cellule
[32]	Sì	Propone una classificazione dei modelli stocastici, descrivendoli
[33]	Sì	Vengono discusse le process algebras nel campo della systems biology
[34]	Sì	Descrive modelli stocastici, un modello agent-based e una sua applicazione
[35]	Sì	Propone il modello agent-based per la modellazione di sistemi biologici e il framework Repast Simphony a esso ispirato

La valutazione della qualità degli studi ha portato a una selezione di 29 documenti dalle libreria digitali. A questi, si aggiungono ulteriori 2 documenti [36, 37], che derivano invece da conoscenze personali, per un totale di 31 testi.

I documenti vengono ora letti attentamente e, per ognuno di essi, si estrapolano le informazioni necessarie a rispondere alla domanda di ricerca, che vengono sintetizzate nel paragrafo successivo.

3 Sintesi delle informazioni

Negli ultimi vent'anni la formulazione di concetti matematici e modelli computazionali è diventata familiare nel campo della biologia, nonostante fosse già applicata molto prima. Basti pensare a Leonardo Fibonacci che, nell'anno 1202, descriveva la crescita di una popolazione di conigli attraverso una successione di numeri, oggi definita come la *successione di Fibonacci*. Con il seguirsi degli anni sono stati introdotti e applicati diversi approcci per una comprensione approfondita della biologia, al fine di mettere in relazione osservazioni biologiche indipendenti tra loro. Vista la complessità della materia, nonché la disponibilità sempre più frequente di nuovi modelli o framework, raggruppare in modo univoco gli approcci di modellazione di sistemi biologici non risulta semplice. Nel 2006, Zoltan Szallasi, Jörg Stelling, Vipul Periwal in [1], riferendosi principalmente al mondo cellulare, propose questa suddivisione:

- **Inferenza bayesiana:** metodo di inferenza statistica in cui il teorema di Bayes viene utilizzato per aggiornare la probabilità di un'ipotesi man mano che più prove o informazioni diventano disponibili

- **Modelli constraint-based:** modelli che incorporano vincoli ben definiti che limitano il comportamento generale della rete

- **Equazioni differenziali ordinarie non lineari:** equazioni che descrivono il tasso di variazione di variabili continue e sono utilizzate principalmente per modellare sistemi dinamici in diverse aree, come i processi cellulari

- **Approcci qualitativi:** comprende metodi che possono fare previsioni di proprietà qualitative della dinamica delle reti di interazione molecolare

- **Modelli stocastici:** modelli che tengono in considerazione le variazioni (causali e non) delle variabili di input, fornendo risultati in termini di probabilità

- **Modelli computazionali:** comprende algoritmi e strutture dati particolarmente adatti nel definire o simulare processi biologici

Nel 2011, gli autori di [8] suddividono ulteriormente i diversi approcci nel seguente modo:

- **Reti booleane:** modellano reti di geni come variabili booleane che possono essere in uno stato attivo o inattivo. A ogni step, lo stato del gene è determinato da una regola logica.

- **Reti bayesiane:** tipologia speciale di grafi probabilistici, in cui i nodi rappresentano variabili casuali e gli archi rappresentano dipendenze condizionali, formando un grafo aciclico diretto.

- **Reti di Petri:** si tratta di grafi bipartiti con due tipi di nodi, posti e transizioni, connessi da archi diretti. I posti possono contenere token, che possono essere consumati quando avviene una transizione di input.

- **Process algebras:** famiglia di linguaggi formali per la modellazione di sistemi concorrenti, tipicamente costituiti da un set di primitive e operatori per la composizione sequenziale e parallela di processi.

- **Modelli *constraint-based*:** l'idea cardine di questo modello è l'incorporamento di vincoli fisico-chimici e biologici ben definiti, che limitano il comportamento generale della rete rispetto ai possibili pattern di flusso.

- **Equazioni differenziali:** descrivono il tasso di variazione di variabili continue. Vengono utilizzate per modellare sistemi dinamici in diverse aree.

- **Modelli *rule-based*:** modelli in cui le specie sono definite in modo strutturato e supportano più stati. Le regole di reazione sono definite come trasformazioni di classi di specie, evitando la necessità di specificare una reazione per ogni possibile stato di una specie. Questa specifica di alto livello viene quindi automaticamente trasformata in una rete biochimica con l'insieme di specie e reazioni generate dalle specifiche.

- **Macchine a stati:** definiscono il sistema in termini di stati piuttosto che di componenti, descrivendo il comportamento temporale di un sistema al cambiamento degli stati delle parti che lo compongono.

- **Automi cellulari:** si tratta di modelli dinamici discreti che consistono in una griglia di celle con un numero finito di stati. Un automa cellulare ha una configurazione iniziale che cambia ad ogni passo temporale attraverso una regola predefinita che calcola lo stato di ogni cella in funzione dello stato dei suoi vicini nello step precedente.

- **Modelli *agent-based*:** descrivono le interazioni tra agenti autonomi e sono utilizzati per studiare fenomeni complessi e dinamiche emergenti usando popolazioni di agenti con regole semplici.

Uno studio più recente, condotto nel 2016 e confluito in [2], suddivide invece gli approcci in tre macro-categorie:

- **Modelli *network-based*:** descrivono e analizzano proprietà, stati o dinamica delle reti. Sono inclusi in questa categoria le equazioni differenziali ordinarie, i modelli stocastici, i modelli booleani e le reti di Petri.

- **Modelli *rule-based*:** comprende tutti quei modelli in cui ogni componente del sistema può aggiornare il suo stato in base a un set di regole.

- **Modelli statistici:** comprende i modelli che stabiliscono relazioni tra i dati misurati e forniscono una guida per estrarre le strutture sottostanti del sistema biologico che ha dato origine ai dati.

Al fine di fornire una descrizione accurata dei diversi approcci di modellazione, essi verranno elencati di seguito e analizzati. Ogni metodo sarà corredato da opportuni riferimenti a uno o più documenti in cui esso viene presentato.

3.1 Le reti booleane

Una rete booleana consiste di un set di variabili booleane $\{\sigma_1, \sigma_2, \ldots, \sigma_n\}$, ovvero variabili che possono assumere solo due valori (0 e 1), il cui valore è determinato da altri variabili nella rete attraverso un set di funzioni booleane $B = \{B_1, B_2, \ldots, B_n\}$, ognuna assegnata a ciascuna variabile. Una rete booleana può essere vista anche come un grafo diretto $G(V, E)$ dove l'insieme dei nodi $V = \{V_1, V_2, \ldots, V_n\}$ corrisponde alle variabili booleane e l'insieme degli archi E è definito implicitamente dalle funzioni booleane nel modello.

Come specificato in [7], le reti booleane forniscono un formalismo efficiente per descrivere le dinamiche di un sistema biologico. Infatti, ogni nodo della rete può essere visto come un componente biologico (ad esempio un gene, una proteina, un metabolita) che è associato a uno stato binario, che specifica se il componente è attivo o meno. Questo modello è adatto per analizzare i comportamenti complessi di un sistema su larga scala come i cambiamenti di attività dei componenti in seguito a perturbazione, le relazioni input-output del sistema e la stabilità delle risposte cellulari a un segnale.

La modellazione di sistemi biologici tramite reti booleane richiede sei passaggi principali [7]:

- **Struttura della rete:** il primo passo consiste nel sintetizzare la struttura della rete raccogliendo in modo estensivo la letteratura pertinente e i dati sperimentali relativi al sistema biologico di interesse.

- **Funzioni di trasferimento booleane:** il secondo passaggio consiste nel determinare le funzioni di trasferimento booleane basate su prove fornite dalla letteratura e osservazioni sperimentali.

- **Transizione degli stati del sistema:** la dinamica di un modello booleano deriva dalla transizione di uno stato di sistema a un altro ed è determinata dalle funzioni di trasferimento ed è influenzata dallo schema di aggiornamento scelto.

- **Analisi dell'output del sistema:** partendo da una condizione iniziale il modello si evolve nel tempo passando da uno stato all'altro e, infine, si stabilizza in un attrattore, che rappresenta il comportamento a lungo termine del sistema. Gli attrattori delle reti corrispondono solitamente agli stati di attivazione costante dei componenti o ai fenotipi cellulari. Pertanto, l'identificazione dei possibili attrattori è utile e biologicamente rilevante.

- **Verifica della correttezza del modello:** un passo importante nella modellazione booleana dei sistemi biologici è testare la correttezza del modello. Esistono diversi modi per convalidare alcune funzionalità del modello. Ad esempio, il modello deve essere in grado di riprodurre precedenti osservazioni sperimentali quali relazioni input-output, comportamenti dinamici e risposte cellulari.

- **Implicazioni e previsioni biologiche:** il vantaggio della modellazione dinamica booleana è la sua capacità di predire i risultati del sistema e generare ipotesi verificabili

I modelli di reti booleane vantano intuitività nella rappresentazione della regolazione genica mediante le funzioni booleane e semplicità degli algoritmi utilizzati per calcolare i grafici di transizione. Tuttavia, i formalismi classici fanno forti ipotesi semplificative, in particolare l'uso di valori binari per l'attivazione genica e le transizioni sincrone. [1]

3.2 PGM: reti di Markov e reti bayesiane

In [14] vengono definiti i PGM, o *Probabilistic Graphical Models*, che possono essere utilizzati per descrivere relazioni dirette o indirette tra variabili. Nei PGM ogni variabile (per esempio, i geni o le proteine) è un nodo nella rete che può essere visto come una variabile casuale, soggetta a incertezza. I collegamenti nella rete trasmettono una misura di associazione rilevante, ad esempio correlazione (non diretta) o causalità (diretta). La struttura della rete può essere scomposta in piccole regioni e tradotta in un prodotto di probabilità condizionali, che rappresenta la distribuzione di probabilità congiunta.

I grafi non diretti sono noti come reti di Markov. Dato un grafo non diretto $G = (V, E)$ e un set di variabili casuali $X = (X_v)_{v \in V}$, si dice rete di Markov se soddisfa le proprietà markoviane:

- Ogni due variabili non adiacenti sono condizionalmente indipendenti, date tutte le altre variabili

- Una variabile è condizionalmente indipendente da tutte le altre variabili dati i suoi vicini

- Qualsiasi due sottoinsiemi di variabili sono condizionalmente indipendenti, dato un sottoinsieme di separazione

Le reti di Markov sono state utilizzate [14] per prevedere la sopravvivenza del cancro al seno dopo che i pazienti hanno ricevuto diverse forme di trattamento, come le combinazioni di chemioterapia, radioterapia e terapia ormonale.

I grafi diretti prendono il nome di reti bayesiane [10, 16]. Una rete bayesiana è definita da:

- **Struttura della rete:** un grafo aciclico diretto $G = (V, A)$ in cui ogni nodo $v_i \in$ V corrisponde a una variabile casuale X_i

- **Distribuzione della probabilità globale:** X con parametri Θ, che può essere scomposta in distribuzioni di probabilità locale più piccole secondo gli archi $a_{ij} \in$ A presenti nel grafo

Il ruolo principale della struttura della rete è di esprimere le relazioni di indipendenza condizionale tra le variabili nel modello attraverso la separazione grafica, specificando così la fattorizzazione della distribuzione globale:

$$P(X) = \prod_{i=1}^{p} P(X_i | \Pi_{X^i}; \theta_{X^i})$$

La distribuzione della probabilità, invece, deve essere scelta in modo tale che la rete bayesiana:

- Possa essere appresa in modo efficiente dai dati
- Sia flessibile
- Sia facile eseguire inferenza tramite query

La flessibilità rende le reti bayesiane particolarmente adatte ad applicazioni biologiche [8]. Infatti, possono essere utilizzate per reti di regolazione genica e *signaling networks*.

Un caso di studio viene proposto in [16] dove i concetti sopra descritti vengono applicati in una *protein signaling network*. Avendo a disposizione 5300 misurazioni simultanee di 11 proteine fosforilate e fosfolipidi, di cui:

- 1800 dati soggetti a stimoli generali
- 600 dati soggetti a stimoli specifici o inibizioni per ognuno delle seguenti quattro proteine: Mek, PIP2, Akt, PKA
- 1200 dati soggetti a stimoli specifici per la proteina PKA

Si vuole apprendere quali relazioni leghino queste 11 proteine. Si sono susseguite tre fasi principali:

- I valori anomali sono stati rimossi e i dati sono stati discretizzati, poiché era impossibile modellarli con una GBN (Gaussian Bayesian Network).

- È stato addestrato un numero elevato di DAG (lipide di membrana facente parte della via di trasduzione del segnale come bersaglio di diversi enzimi) per produrre un modello più robusto.

- La validità della rete bayesiana è stata valutata rispetto ai *signalling pathways* dalla letteratura

Uno degli svantaggi delle reti bayesiane è l'impossibilità di modellare *feedback loops*, che invece sono ricorrenti nelle reti biologiche. Questa limitazione può essere superata tramite reti bayesiane dinamiche, in cui le variabili sono replicate per ogni step e il feedback è modellato connettendo i nodi a intervalli temporali adiacenti. [8]

3.3 Le reti di Petri

Le reti di Petri [8, 15, 17] sono un approccio per rappresentare sistemi concorrenti, ideate da Carl Adam Petri e, successivamente, adattate ed estese in molte direzioni. La versione più semplice è costituita da un grafo bipartito, ovvero un grafo con due tipi di nodi e archi direzionali che connettono i diversi nodi. Quest'ultimi sono chiamati *places*, rappresentati come cerchi e *transitions*, rappresentati come riquadri. Gli archi possono essere etichettati con un valore intero che ne definisce il peso. I *places* possono essere contrassegnati da un numero intero di *tokens*. Gli archi che connettono a un transition node definiscono un set di places di input e output per quella transizione. Una transizione è abilitata solo se il numero di token è uguale o maggiore del peso dell'arco che connette i due places.

Nel corso degli anni sono state proposte diverse estensioni:

- **Rete di Petri gerarchica:** permette la composizione di relazioni, dove una precedente rete è rappresentata da un place oppure una transizione nella nuova rete

- **Rete di Petri ibrida:** incorpora places che possono ottenere valori continui di token anziché numeri interi

- **Rete di Petri timed:** nella quale ai places o alle transitions è possibile assegnare valori deterministici di tempo corrispondenti a ritardi

- **Rete di Petri stocastica:** nella quale ai places e transitions è assegnato un valore di ritardo che è dato dalla distribuzione di probabilità

- **Rete di Petri colorata:** che consente ai token di avere una struttura e transizioni interne al fine di avere regole di *firing* più complesse

Le reti di Petri sono state utilizzate in diversi studi di sistemi biologici, sia per analisi qualitative sia quantitative. Per esempio, in una rete di Petri i places sono utilizzati per rappresentare entità biochimiche (enzimi, ioni, ecc.) e le transitions per definire le reazioni. Inoltre, tramite opportune estensioni della rete è possibile modellare archi speciali che modellano la catalisi o l'inibizione.

Un'estensione della rete di Petri ibrida è stata utilizzata nella rappresentazione e simulazione di reti di regolazione genica e percorsi di trasduzione del segnale. Questo modello è stato implementato come software dal nome *Genomic Object Net*, che consente all'utente di studiare l'evoluzione temporale delle concentrazioni di specie all'interno di un sistema. Le reti di Petri colorate, invece, sono state utilizzate per simulare l'evoluzione temporale delle concentrazioni di metaboliti in un percorso metabolico.

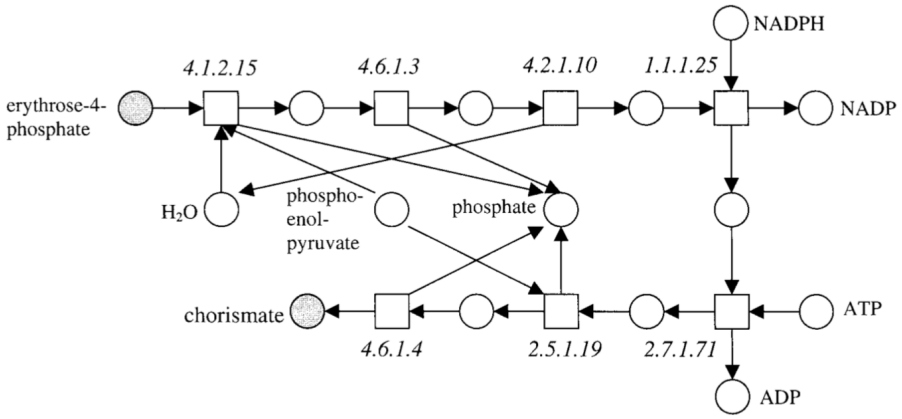

Figura 1. Rappresentazione dello *shikimate pathway* tramite rete di Petri: i places rappresentano i metaboliti mentre le transitions rappresentano le reazioni.

3.4 Equazioni differenziali

Le equazioni differenziali descrivono il tasso di variazione di variabili continue e sono utilizzate principalmente per modellare sistemi dinamici in diverse aree [3]. I processi cellulari, ad esempio [1], sono controllati da complesse reti di macromolecole che interagiscono tra di loro. Queste reti sono insiemi di reazioni chimiche che convertono le specie reagenti in specie di prodotto a velocità che dipendono dalle concentrazioni dei reagenti e, spesso, dalle concentrazioni di altre molecole. Una rete di questo tipo può essere descritta matematicamente come un set di equazioni differenziali ordinarie (ODE) non lineari, che tengono traccia degli effetti di queste reazioni che occorrono simultaneamente.

Nel realizzare una rete come un sistema di ODE occorre soffermarsi su due assunzioni. Innanzitutto, il nostro sistema deve essere un reattore chimico *well-stirred*, ovvero le concentrazioni dei componenti non variano rispetto allo spazio. In secondo luogo, le variabili (concentrazioni chimiche) sono funzioni continue del tempo.

Nello sviluppare il set di ODE, occorre immaginare la rete come un sistema dinamico il cui lo stato cambia da un istante di tempo all'altro. Assegniamo a ogni specie nel diagramma una singola variabile di stato, *X(t)*, che rappresenta la concentrazione di specie X. La collezione dei valori di tutte queste variabili $\{X_1(t),\ X_2(t),\ X_3(t),\ ...\}$ costituisce lo stato del sistema. In seguito, per ogni specie molecolare si scrive un'equazione differenziale che descrive come la sua concentrazione cambi nel tempo a seguito di interazioni con altre specie nella rete.

$$\frac{dX}{dT} = sintesi - degradazione - fosforilazione + defosforilazione, ecc.$$

Ogni reazione (sintesi, degradazione, ecc.) deve essere rappresentata da una legge cinetica, la quale avrà una o più costanti associate ad essa. Assegnando specifici valori a queste costanti, perfezioniamo le leggi a reazioni particolari. L'insieme di tutte le costanti necessarie per descrivere le reazioni in una rete di interazione molecolare è chiamato *parameter set* $\{p_1,\ p_2,\ ...\ ,\ p_m\}$ del modello. Con questo paradigma le conseguenze dinamiche della rete sono determinate dal sistema di equazioni differenziali ordinarie non lineari:

$$\frac{dX_i}{dT} = F_i(X_1, X_2, ..., X_n; p_1, p_2, ..., p_m), i = 1, 2, ..., n$$

Un'altra possibilità di modellazione dei sistemi biologici tramite equazioni differenziali è data dalle equazioni alle derivate parziali (PDE), particolarmente adatte per modellare i sistemi con variabili multiple come tempo e spazio [18, 24]. Una PDE descrive la funzione indirettamente attraverso una relazione fra se stessa e le sue derivate parziali, invece di scrivere esplicitamente la funzione. La relazione deve essere locale, cioè deve connettere la funzione e le sue derivate nello stesso punto. Un esempio di applicazione viene fornito da Dilão in *mRNA diffusion explains protein gradients in Drosophila early development*, dove si specifica il modello di diffusione del mRNA nel seguente modo:

$$\frac{\delta X}{\delta T} = s - dX + D\frac{\delta^2 X}{\delta x^2},$$

$$\frac{\delta X}{\delta T} = aX$$

dove X è la concentrazione di proteina, A è il tasso di produzione di mRNA, d è il tasso di degradazione di mRNA nel citoplasma.

In [14] si specifica come le ODE possano essere applicate per studiare le proprietà dinamiche del cancro. Ad esempio, un modello di tumore soppressore p53 e oncogene Mdm2 ha rivelato un'elevata variabilità nei comportamenti oscillatori delle cellule dopo il danno al DNA.

3.5 Automi cellulari

Gli automi cellulari [8] sono modelli discreti che consistono di una griglia di celle con un numero finito di stati. Un automa cellulare ha una configurazione iniziale che cambia ad ogni step di tempo attraverso una regola predefinita che calcola lo stato di ogni cella in funzione dello stato dei suoi vicini nello step precedente. Gli automi cellulari sono stati utilizzati per lungo tempo in studi di Biologia. Uno degli esempi più noti è l'*excitable media*, che imita il comportamento delle cellule del sistema nervoso, dei muscoli e le funzioni cardiache. Grazie alle loro caratteristiche spaziali, gli automi cellulari sono stati utilizzati principalmente per studiare dinamiche molecolari e della popolazione cellulare.

Possiamo suddividere gli automi cellulari in tre gruppi principali:

- **Automi deterministici:** il dominio spaziale del modello è diviso in un reticolo prefissato e ogni punto del reticolo è associato a uno stato. Lo stato di una cella allo step successivo è determinato solamente dallo stato delle cellule vicine nello step precedente.

- **Modelli a gas reticolato:** consiste di una griglia spaziale discreta nella quale le particelle si muovono e interagiscono in un modo prestabilito. A differenza degli automi deterministici, queste sistemi sono guidati spesso da eventi casuali.

- **Modelli "solidification"**: simile al modello precedente, eccetto che una volta che una particella si trova in uno stato *bound*, non può più muoversi né scomparire.

3.6 Macchine a stati

Le macchine a stati [8, 15] sono formalismi basati su diagrammi che descrivono il comportamento temporale di un sistema in base ai cambiamenti negli stati delle sue parti. Sono adatti per modellare il comportamento biologico in modo qualitativo poiché richiedono pochi dati quantitativi. Differiscono dagli altri approcci in quanto definiscono un sistema in termini di stati piuttosto che di componenti. Sono tipicamente utilizzati per il model checking e l'esecuzione interattiva.

Un primo formalismo di questa tipologia, gli *statecharts*, è stato sviluppato da David Harel negli anni '80 ed è stato applicato per la prima volta in Biologia per la modellizzazione del processo di attivazione delle cellule T e, più recentemente,

nell'organogenesi pancreatica. In questo formalismo, lo stato di un sistema può contenere sotto-stati a più livelli, consentendo una visione gerarchica del sistema e una relazione tra gli eventi su scale sempre più grandi.

Altri formalismi correlati sono *Reactive Modules* e *Live Sequence Charts* che, insieme al primo, sono stati applicati alla modellizzazione dello sviluppo di C. elegans vulva.

3.7 Modelli stocastici

Molti modelli biologici assumono che le dinamiche osservate siano guidate esclusivamente da meccanismi interni deterministici. Tuttavia, i sistemi biologici reali sono esposti a fenomeni che non sono completamente definibili esplicitamente da un modello. Occorre, quindi, estendere i modelli deterministici a modelli che accolgono variazioni più complesse nelle dinamiche inserendo, per esempio, approcci stocastici. [34]

Una possibilità [10, 19] è estendere le classiche equazioni differenziali tramite un sistema di equazioni differenziali stocastiche (SDE), dove i parametri sono randomizzati o modellati come processi casuali. Questo approccio assume che vi sia un certo grado di disturbo nelle dinamiche dei processi. Ne consegue un sistema con parti sia deterministiche sia stocastiche, definibile nel seguente modo:

$$dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_t$$

dove $\{X_t = X(t)\}_{t \geqslant 0}$ è un processo stocastico.

In [32] i modelli stocastici vengono suddivisi in due gruppi principali:

- **Non-spatial models**: si basano sull'assunzione che i processi di miscelazione avvengano più velocemente dei processi di reazione
- **Spatial models**: si occupano dell'organizzazione spaziale delle proteine e delle membrane in modo esplicito

Nel primo gruppo ha un ruolo centrale la *Chemical Master Equation*, che descrive l'evoluzione nel tempo di $p(x, t)$, ovvero la probabilità che il sistema rimarrà nello stato x al tempo t. [32, 5]

$$p(x,t) = -p(x,t)\sum_{\mu=1}^{M} a_\mu(x) + \sum_{\mu=1}^{M} p(x - s_\mu, t)a_\mu(x - s_\mu)$$

Essa è stata utilizzata per studiare la dinamica dei segmenti denaturati di DNA a doppio filamento. Inoltre, studi su motori molecolari tramite CME hanno dimostrato come la curva carico-velocità provenga da energie libere che legano

trifosfato di nucleotide. Nel 1976 Gillespie ha proposto un algoritmo per simulare la CME, noto come *algoritmo di Gilliespie*. Questo algoritmo si compone di tre fasi principali:

- Genera un time step fino alla successiva reazione
- Determina la tipologia di reazione
- Esegue la reazione facendo avanzare il conteggio del tempo e delle molecole

Il secondo gruppo, invece, si basa sull'idea che molti sistemi biologici siano di gran lunga organizzati. Un esempio è dato dagli eucarioti, che hanno organelli elaborati, microtubuli e altri elementi complessi del citoscheletro, proteine motorie che si muovono avanti e indietro e forme cellulari controllate attentamente. Simulazioni spaziali sono state utilizzate per investigare diversi topic di carattere biologico, tra cui figura: gradienti di morfogenione tra gli ovociti di Drosophila e Xenopus, il sistema di localizzazione del piano di divisione cellulare di Escherichia coli e segnalazione intracellulare.

3.8 Modelli constraint-based

L'idea alla base dei modelli constraint-based [1, 8] è l'incorporamento di vincoli fisico-chimici e biologici ben definiti che limitano il comportamento generale della rete rispetto ai possibili schemi di flusso. Si basano su vincoli di capacità stechiometrici, termodinamici ed enzimatici e, invece di una singola soluzione, definiscono uno spazio di possibili soluzioni che rappresentano fenotipi diversi conformi a vincoli. Questi modelli sono stati utilizzati per la determinazione delle distribuzioni di flusso, analisi del bilancio di flusso e pronostici di fenotipo knockout.

3.9 Modelli rule-based

I modelli rule-based [15] hanno ottenuto molta attenzione tra i biologi in quanto la loro notazione è molto simile alla rappresentazione delle reazioni chimiche, utilizzate nella systems biology per modellare le interazioni biochimiche tra specie molecolari.

Consideriamo, per esempio, la classica reazione enzimantica dove un enzima (E) si lega a un substrato (S) e produce un prodotto (P) rilascaindo l'enzima (E). Questo può essere espresso in modo compatto e coinciso utilizzando due semplici regole:

$$E + S \rightleftharpoons ES$$
$$ES \rightarrow E + P$$

Le regole, rispetto alle equazioni, hanno la caratteristica di essere unità indipendenti, che possono essere facilmente cambiate o modificate. Inoltre, la sintassi semplice dei modelli rule-based può essere memorizzata in un file human-readable e può essere modificata e visualizzata tramite una rappresentazione grafica.

3.10 Process algebras

La famiglia delle *process algebras* ha alcune proprietà interessanti che la rendono particolarmente utile nella descrizione di sistemi biologici. Innanzitutto [15], le process algebras offrono composizionalità, ovvero la possibilità di definire l'intero sistema a partire dalla definizione dei suoi sottocomponenti. In secondo luogo, forniscono una rappresentazione formale del sistema evitando ambiguità. Permettono, infine, di astrarre i sistemi biologici da sistemi concorrenti descritti tramite process algebras: le specie possono essere viste come processi in grado di interagire tra loro e le reazioni possono essere modellate usando azioni. Le process algebras più note [30] sono Milner's Calculus of Communicating Systems (CCS) e Hoare's Communicating Sequential Processes (CSP).

Recentemente, come risposta alla necessità di modellare le dinamiche di sistemi biologici complessi, ci sono state diverse applicazioni di process algebra nei sistemi biologici. Queste tecniche sono appropriate per descrivere e analizzare formalmente un sistema biologico nel suo insieme e per effettuare analisi sulle interazioni proteina / gene. Esse possiedono alcune proprietà che le rendono particolarmente adatte nella modellazione di sistemi biologici:

- Permettono la specificazione formale di un sistema
- Forniscono diversi livelli di astrazione per lo stesso sistema biologico
- Possono rendere le interazioni esplicite, ad esempio gli elementi biologici possono essere visti come entità che interagiscono ed evolvono
- Supportano modularità e composizionalità
- Forniscono tecniche consolidate per analizzare determinati comportamenti

In [33] vengono individuati due gruppi di *process calculi* all'interno delle process algebras:

- Process calculi che derivano dall'informatica e sono state applicate in seguito alla biologia, come π-calculus, CCS-R e PEPA

- Process calculi definiti dall'osservazione di strutture e fenomeni biologici, come Beta-Binders e k-calculus

π-calculus. È un modello matematico di processi le cui interconnessioni cambiano nel momento in cui interagiscono. Lo step base è il trasferimento di un *communication link* tra processi. Il destinatario può utilizzare questo link per ulteriori interazioni con altre parti. Queste caratteristiche rendono il π-calculus particolarmente adatto per modellare sistemi dove le risorse accessibili si modificano nel tempo. L'idea alla base dell'applicazione del π-calculus alla biologia è l'astrazione *molecule-as-computation*: ogni entità e interazione biologica è associata a una specifica dell'agente nel calcolo. Nello specifico, le molecole sono modellate come processi, le capacità di interazione come canali, le interazioni come comunicazioni tra processi, le modifiche come cambiamenti di stato e canale e, infine, i compartimenti e le membrane come restrizioni.

CCS-R. Variante di CCS con nuovi elementi che permettono reversibilità. Le interazioni sono descritte come comunicazioni binarie sincronizzate.

PEPA. È una SPA (Stochastic Process Algebra) per la modellazione di sistemi con comportamento concorrente. In PEPA ogni azione è associata a una durata, rappresentata da una variabile casuale con una distribuzione esponenziale negativa. PEPA è stata applicata per modellare e analizzare *signalling pathways*. Un ulteriore studio si focalizza sull'influenza di RKIP (Raf Kinase Inhibitor Protein) su ERK (Extracellular Regulated Kinase). PEPA è adatta per sistemi biologici in quanto fornisce un alto livello di astrazione per il modello e si focalizza sulla composizionalità e sulle interazioni.

Beta-Binders. Sono un'estensione del π-calculus, ispirata dai fenomeni biologici. Questo process calculus si basa sul concetto di *bio-process*, un contenitore di siti che esprimono le capacità di interazione dell'elemento. I Beta-binders estendono il π-calculus con alcuni costrutti che permettono di rappresentare features biologiche, come l'unione di due bio-process, la scissione di un bio-process in due o cambiamenti nella loro interfaccia.

***k*-calculus.** Modello formale utilizzato per le interazioni delle proteine. Le unità del modello sono le proteine e gli operatori definiscono la creazione e la divisione di complessi proteici. Le proteine sono disegnate come *boxes* con i siti nel loro bordo, che possono essere visibili o nascosti. Oltre alla rappresentazione grafica, il k-calculus fornisce un linguaggio nello stile delle process algebras. Alle espressioni e ai boxes viene data semantica da una serie di reazioni di base. Una volta specificato il sistema iniziale, il comportamento del sistema si ottiene riscrivendolo tramite regole di riduzione.

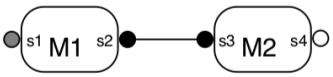

Figura 2. Esempio di rappresentazione tramite k-calculus. M1 e M2 sono due molecole delimitate sui siti s2 e s3. Inoltre, il sito s1 è nascosto mentre il sito s4 è visibile.

In [26] viene proposto *Hype* come process algebra per descrivere sistemi ibridi, ovvero sistemi che mostrano sia un comportamento continuo sia uno discreto. Hype si differenzia da altre process algebra in quanto non richiede una visione monolitica dei sottosistemi e, di conseguenza, non è necessario che il comportamento continuo di un sottosistema sia compreso prima del processo di modellazione. In Hype abbiamo tre livelli di semantica: la semantica operazionale, che descrive il comportamento del sistema in termini di eventi discreti; la semantica ibrida,

che utilizza tale comportamento per ottenere le equazioni differenziali ordinarie; infine, la semantica dell'equivalenza, che descrive nozioni di comportamento simile. Hype può essere utilizzato per rappresentare reti di regolazione genetica sintetica, come il *Repressilator*.

In [25] viene proposto un linguaggio per costruire modelli di sistemi biologici basati su process algebra, nello specifico π-calculus. L'idea del linguaggio è che esistano set di specie { A, B, C, ... }, ognuno dei quali ha un numero di siti { a, b, c, ... } con cui può legarsi ad altre specie o non legarsi, qualora siano già legati. È possibile definire frasi che specificano il comportamento delle specie in relazione ai loro siti nel seguente modo:

$$< type, (A, a), (B, b), Pos, Neg, r >$$

dove type (che può assumere i valori di associazione, dissociazione, trasformazione) rappresenta il tipo di frase. Le coppie (A, a) e (B, b) sono chiamate *body* mentre *Pos* e *Neg* sono le condizioni della frase.

Questo modello viene, in seguito, tradotto in *compile map*, un set di espressioni che prendono il nome di *process description* per ogni specie del modello. Infine, per ogni process description viene creata una specifica di processo nel π-calculus stocastico.

In [31] viene proposto bio$k$-calculus, un'alternativa al k-calculus per descrivere proteine e cellule. Il core di bio$k$-calculus ha la seguente sintassi:

$$S ::= 0 | A(\sigma) | M(|S|)[S] | S, S$$

La sintassi definisce l'insieme delle soluzioni S, che possono essere vuote, composte da una proteina A(σ) o una cellula M(|S|)[S] oppure un gruppo di soluzioni S, S. Le reazioni considerate in questo linguaggio sono principalmente due: complessazioni, che creano spigoli tra le proteine, eventualmente disconnesse e le decomplessazioni, che rimuovono i bordi.

3.11 Linguaggi di Markup

Con linguaggio di markup si intende un linguaggio che permette di descrivere i dati attraverso una formattazione specifica che utilizza marcatori, detti anche tag. In [20] e [21] vengono proposti due linguaggi di markup che possono essere utilizzati nella modellazione di sistemi biologici, rispettivamente SBML (Systems Biology Markup Language) e PHML (Physiological Hierarchy Markup Language).

SBML è un linguaggio XML-based che permette di suddividere una reazione chimica in un numero di elementi concettuali. Nello specifico, un modello in SBML consiste di una lista con uno o più dei seguenti componenti:

- **Compartimento:** un contenitore di volume finito dove le reazioni hanno luogo
- **Specie:** entità chimica che prende parte alla reazione
- **Reazione:** dichiarazione che descrive alcuni processi di trasformazione, trasporto o associazione che possono cambiare una o più specie
- **Parametro:** una quantità che ha un nome simbolico, che può essere globale in un modello oppure locale per una singola reazione
- **Definizione di unità:** un nome per l'unità usata nelle espressione delle quantità in un modello
- **Regola:** un'espressione matematica che è aggiunta alle equazioni del modello costruita dal set di reazioni

Come si evince dal frammento di codice, lo scheletro di un modello SBML rispecchia le caratteristiche standard in un documento XML: testo in cui ogni elemento è una coppia di tag racchiusi dai caratteri < e > e alcuni di essi contengono attributi nella forma *Attribute='value'*. L'elemento *sbml* incapsula il modello SBML con le caratteristiche elencate sopra.

```
<?xml version="1.0" encoding="UTF-8"?>
<sbml xmlns="http://www.sbml.org/sbml/level1"
      level="1" version="2">
  <model name="gene_network_model">
    <listOfUnitDefinitions>
        ...
    </listOfUnitDefinitions>
    <listOfCompartments>
        ...
    </listOfCompartments>
    <listOfSpecies>
        ...
    </listOfSpecies>
    <listOfParameters>
        ...
    </listOfParameters>
    <listOfRules>
        ...
    </listOfRules>
    <listOfReactions>
        ...
    </listOfReactions>
  </model>
```

```
</sbml>
```

Listing 1.1. Scheletro della definizione di un modello espresso in SBML

PHML è anch'esso un linguaggio XML-based, utilizzato per descrivere modelli biofisici multilivello. In PHML, ciascuno degli elementi che costituiscono il modello è chiamato *modulo* e le relazioni strutturali e funzionali tra moduli sono definite da *spigoli*. Ogni modulo è quantitativamente caratterizzato da diverse variabili dinamiche, costanti, parametri time-dependent e dati di morfologia, che sono definiti come *quantità fisiche* in PHML. La definizione delle dinamiche o delle funzioni delle quantità fisiche è esplicitamente descritta da equazioni matematiche. Ci sono due tipi di spigoli in PHML: uno funzionale e l'altro strutturale.

La definizione di una relazione funzionale tra due moduli è rappresentata da un *functional edge* che collega un out-port di un modulo a un in-port di un altro modulo, portando il valore della quantità fisica associata all'out-port del modulo precedente all'altro modulo. Il modulo riceve il valore dalla in-port da un'altro modulo e lo memorizza nella quantità fisica assegnata a quella in-port. Successivamente il modulo può utilizzare il valore proveniente dall'esterno nelle equazioni definite in esso.

Una struttura logica, invece, rappresenta la relazione ontologica tra moduli come una relazione *has-a*. In termini di fisiologia, corrisponde al *constitute* (esempio: molti cardiomiociti costituiscono un cuore) e all'*include* (esempio: una membrana cellulare include organelli).

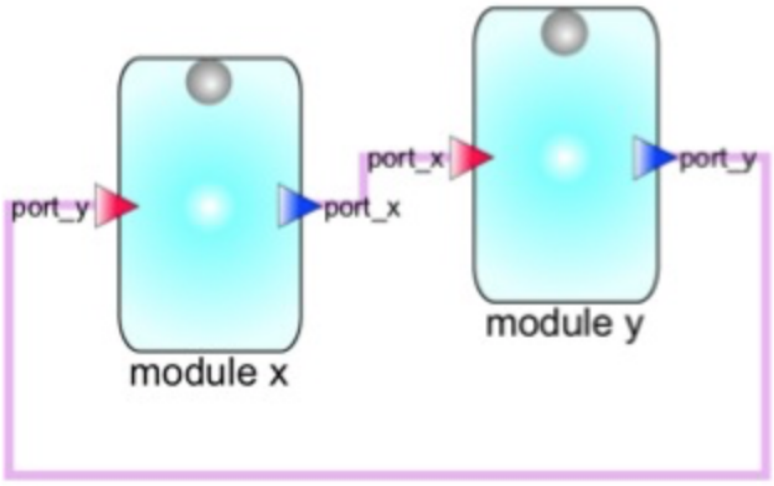

Figura 3. Rappresentazione di un modello in PHML

3.12 Membrane computing

Membrane computing è una branca dell'informatica che concettualizza idee e modelli di calcolo basati sulla struttura e sul processo delle cellule viventi. In [22] viene descritto come un'alternativa al sistema di ODE, inserendo quelle feature che possono essere utilizzate in applicazioni biologiche. In particolare:

- **Capacità di distribuzione:** la composizione dei comportamenti locali consente l'interazione non lineare della parte di sistema e il comportamento emergente

- **Natura discreta:** i processi biologici sono discreti e vengono offerti come alternativa al sistema di equazioni differenziali

- **Rappresentazione algoritmica:** membrane computing è un modello computazionale, lo stesso delle rappresentazioni degli algoritmi di Turing e, quindi, facilmente simulabile sui computer

- **Scalabilità/Estensibilità:** modularità disponibile in membrane computing per contrastare le difficoltà nelle equazioni differenziali

- **Trasparenza:** le regole di riscrittura multiset in membrane computing sono equazioni di reazione senza alcuna notazione o comportamento complesso, come utilizzato in biochimica

- **Massiccio parallelismo:** comportamento generale di un sistema biologico

- **Non determinismo:** un insieme di regole localizzate in determinati compartimenti, senza alcun ordine imposto nell'implementazione delle regole

- **Comunicazione:** interazione tra i compartimenti

Membrane computing è principalmente utilizzato per modellare i seguenti sistemi biologici:

- Sistemi biologici coinvolti nelle reazioni molecolari, nei cambiamenti dell'ambiente o cambiamenti causati da stimolazioni esterne di una certa pressione o stimolo specifico

- Sistemi biologici che danno la priorità all'esecuzione di processi interconnessi in sequenza, in cui un processo deve essere completato prima che venga eseguito il successivo

- Sistemi biologici che coinvolgono comportamenti emergenti, in cui le interazioni tra processi all'interno di un compartimento o interazioni tra compartimenti generano un comportamento globale

3.13 Multiset Rewriting

Multiset Rewriting (MSR) è un formalismo logic-based che si fonda sulla riscrittura di sistemi [29]. MSR offre sia un linguaggio formale, per descrivere mappe di interazioni molecolari sia un modello di esecuzione, che permette la simulazione delle dinamiche delle reti molecolari. I componenti principali del linguaggio sono i *multiset* e le *regole di riscrittura*.

Sia $\sum$ un set finito di nomi di specie. Un multiset è una funzione $s : \sum \mapsto \mathbb{N}$ e $\delta(\sum)$ rappresenta l'universo dei multiset su $\sum$. Un multiset rappresenta la configurazione di un sistema biologico e i possibili eventi sono modellati tramite *rewriting rule*.

Una regola di riscrittura è una coppia $R = (l, r)$, dove $l \in \delta(\sum)$ e $r \in \delta(\sum)$ sono multiset, chiamati reagenti e prodotti rispettivamente.

Grazie alla sua semplicità ed espressività, MSR è in grado di descrivere diversi sistemi biologici di interesse.

3.14 Modelli agent-based

I modelli agent-based descrivono le interazioni che avvengono tra diversi agenti autonomi. Un agente è un'astrazione software di alto livello [4, 8] che fornisce una modalità per descrivere un'entità software complessa in termini del suo comportamento all'interno di un ambiente computazionale. Esistono due tipologie di definizione:

- **Weak**: gli agenti vengono descritti come entità computazionali reattive (rispondono a cambiamenti che avvengono nell'ambiente), proattive (mantengono gli obiettivi generali), autonome (non sono controllate esternamente) e interagiscono con altre entità.

- **Strong:** gli agenti vengono descritti come entità composte da particolari abilità mentali o cognitive, suggerendo architetture basate sul modello BDI (belief - desire - intention)

In quest'ottica ha un ruolo chiave il MAS (Multi-Agent System) - nel quale più agenti interagiscono in un'architettura di sistema globale - in quanto ha un livello di astrazione tale da permettere la risoluzione di problemi complessi. Questo è possibile grazie alla modellizzazione e ingegnerizzazione di sistemi complessi, che sono caratterizzati da strutture di organizzazione e processi di coordinazione che sono più articolati e più dinamici rispetto a forme alternative.

Dette queste premesse, possiamo considerare un agente un'entità in grado di eseguire azioni di *problem solving* flessibili e autonome e di operare come un processo stand-alone, eseguendo operazioni senza l'intervento dell'utente, mantenendo una descrizione del suo stato e dello stato dell'ambiente nel quale si

trova. Poiché l'ambiente in cui è immerso l'agente è tipicamente dinamico, aperto, imprevedibile e popolato da altri agenti, è necessario che gli agenti siano in grado di comunicare tra di loro e con l'ambiente di esecuzione stesso.

In [23] viene presentato ELECANS per lo sviluppo di sistemi biologici complessi, in particolare per lo studio del cancro. È costituito di tre componenti principali:

- **ELECANS SDK**: il core della piattaforma, costituito da un engine multi-agente per processare agenti che definiscono cellule e il loro comportamento
- **Rules Editors:** permette di definire alcune regole, in particolare regole per la cellula o regole di simulazione
- **Tissue Geometry Editor:** permette di definire gruppi di cellule e geometrie
- **Third Party .NET Bridges:** permette a ELECANS di accedere ed eseguire modelli scritti in linguaggi differenti, come C.

In [35] viene proposto, invece, Repast (Recursive Porous Agent Simulation Toolkit) Simphony come framework per la simulazione e modellazione agent-based. Sviluppato in Java, utilizza Eclipse come ambiente di sviluppo integrato per produrre codice. I tools di Repast Simphony permettono ai ricercatori di sviluppare modelli in modo flessibile, grazie a una GUI, una toolbar per controllare i processi di simulazione (start, step, pause, stop, exit), alla possibilità di visualizzare gli agenti e il loro ambiente, monitorando i dati di output.

Per creare un modello agent-based in Repast Simphony, occorre creare delle classi. Possono essere create diverse classi per un certo scenario del modello che includono un certo numero di metodi per descrivere gli attributi e i ruoli degli agenti. I metodi di *setup* e di *step* vengono richiamati per ogni iterazione della simulazione e il *simulation runtime* è definito da step di tempo o conteggio di tick. Durante l'esecuzione della simulazione, gli agenti possono svolgere delle azioni. I metodi di *get* e *set*, che descrivono gli attributi degli agenti, possono aggiornare il valore memorizzato in ogni conteggio di tick. Gli agenti, inoltre, possono aggiornare le loro azioni in base ai risultati delle precedenti azioni svolte.

Repast Simphony può essere applicato in diverse simulazioni, ad esempio il modello preda-predatore, la popolazione batterica e la resistenza agli antibiotici e l'omeostasi.

In [36] e [37] si descrive lo sviluppo di sistemi multi-agente tramite AgentSpeak e Jason. AgentSpeak è un linguaggio di programmazione per agenti BDI, basato sulla programmazione logica. I principali costrutti di AgentSpeak sono:

- **Beliefs:** definiscono lo stato corrente dell'agente e informazioni riguardo all'ambiente
- **Goals:** definiscono lo stato che l'agente vuole raggiungere
- **Plans:** rappresentano il *know-how* dell'agente

Un piano in AgentSpeak ha la seguente struttura:

```
triggering_event : context <- body.
```

dove:

- **triggering_event:** indica gli eventi che il piano intende gestire
- **context:** definisce le circostanze nelle quali il piano può essere utliizzato
- **body:** flusso di azioni da utilizzare per gestire l'evento se il contesto è *true* nel momento in cui il piano viene scelto per gestire l'evento

I *triggering events* disponibili in AgentSpeak sono:

- +**b** (aggiunta di un belief)
- −**b** (rimozione di un belief)
- +**!g** (aggiunta di un achievement-goal)
- −**!g** (rimozione di un achievement-goal)
- +**?g** (aggiunta di un test-goal)
- −**?b** (rimozione di un test-goal)

Jason implementa le semantiche operazionali di una variante di AgentSpeak, estendendolo, in modo tale da essere un linguaggio per definire agenti. Le estensioni permettono di:

- Utilizzare un linguaggio di alto livello per definire il comportamento degli agenti (goal oriented)
- Utilizzare Java come linguaggio di basso livello per realizzare meccanismi e modificare l'architettura

Dall'architettura di un agente Jason, mostrata nella figura, possiamo definire il *reasoning cycle* di Jason, o steps:

- Percezione dell'ambiente
- Aggiornamento del belief base
- Ricezione di comunicazioni da altri agenti
- Selezioni di messaggi "socialmente accettabili"
- Selezione di un evento
- Recupero di tutti i piani rilevanti
- Determinazione dei piani applicabili
- Selezione di un piano applicabile

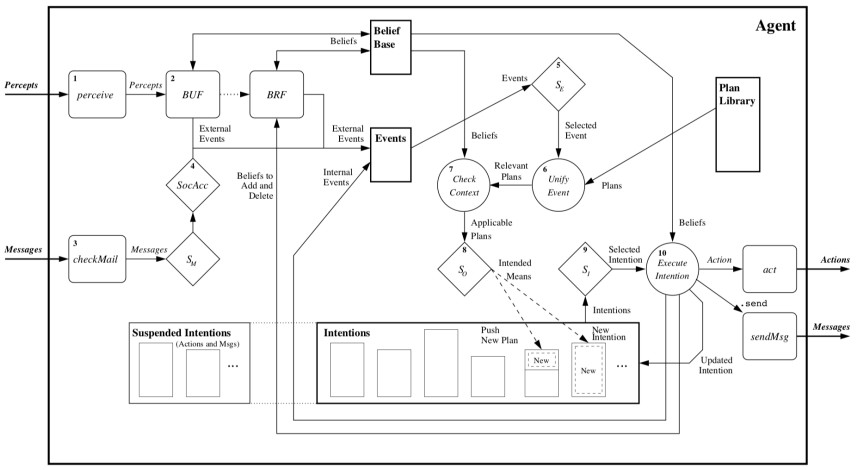

Figura 4. Reasoning Cycle in Jason

– Selezione di una intention per ulteriori esecuzioni
– Esecuzione di uno step della intention

Possiamo utilizzare Jason per implementare modelli di sistemi biologici, per esempio gli automi cellulari. Nell'esempio, allegato a questa SLR, si propone il *cyclic cellular automaton*.

Il cyclic cellular automaton è un automa cellulare che si basa sul seguente set di regole:

– Si definisce una griglia 2D di cellule
– Si definisce un numero massimo di stati che la cellula può avere
– Si definisce un valore di soglia
– Si associa ad ogni cellula un valore compreso tra 0 e il numero massimo di stati - 1, rappresentante lo stato della cellula
– A ogni step della simulazione ogni cellula segue le seguenti regole:

  • Si contano il numero di cellule vicine (tramite regola di Moore o Von Neumann) che circondano la cellula corrente e che hanno un valore di stato incrementato di 1 rispetto a tale cellula
  • Se il conteggio è maggiore o uguale al valore di soglia, lo stato della cellula corrente viene incrementato di 1

– Si ripete il processo

In una visione ad agenti, possiamo considerare ogni singola cellula come un agente autonomo, immerso nell'ambiente. L'agente è in grado di percepire l'ambiente

che lo circonda (in questo caso, le cellule vicine) e cambiare eventualmente il proprio stato (qualora il numero di vicini con determinate caratteristiche sia maggiore di un certo valore di soglia). Il cambiamento di stato di un agente potrebbe riflettersi successivamente nelle cellule vicine, che potrebbero cambiare a loro volta il loro stato. Possiamo riassumere quanto detto con la seguente figura:

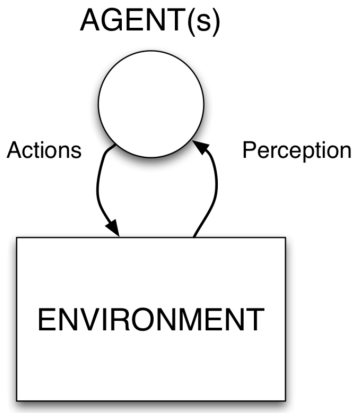

Figura 5. Interazione tra agenti e l'ambiente

La funzione *perception* definisce l'abilità dell'agente di osservare l'ambiente in cui è immerso mentre la funzione *action* rappresenta il processo decisionale dell'agente.

Possiamo definire un agente in Jason tramite un file *.asl*, nel quale è possibile specificare i beliefs e i goal iniziali, nonché i piani. Riferendoci all'esempio proposto, il nostro agente verificherà l'esistenza di circostanze che permettano di utilizzare il piano (ovvero che il numero di cellule vicine con uno stato uguale allo stato della cellula corrente incrementato di 1 sia maggiore di una determinata soglia, ad esempio 4) e, in caso affermativo, procede all'esecuzione di un'azione che, in questo caso, modifica lo stato corrente della cellula, incrementandolo. Viceversa, l'agente non deve fare nulla, mantenendo lo stato attuale.

```
/* Plans */

+next(_) : neighbours_with_upper_value(N) & N >= 4 <- inc.

+next(_) <- doNothing.
```

Listing 1.2. Descrizione dei piani dell'agente rappresentante una cellula

Come detto in precedenza, l'agente si trova immerso in un ambiente. In Jason è possibile implementare un ambiente estendendo la classe *Environment* e facendo override dei metodi *init* e *executeAction*.

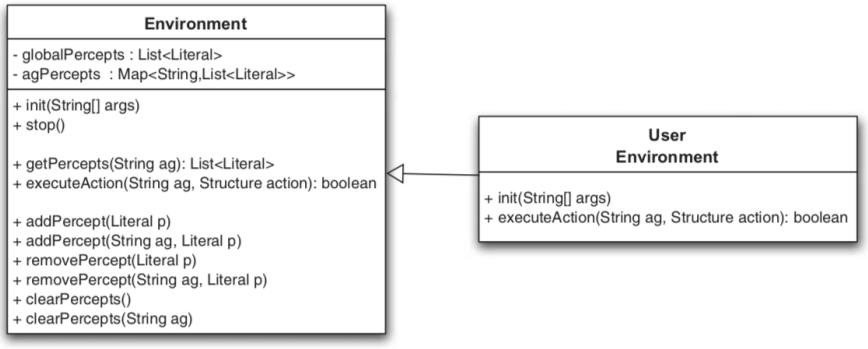

Figura 6. Implementazione di un environment da un'applicazione utente

executeAction è uno dei metodi più importanti, in quanto si definiscono in esso gli effetti delle azioni dell'agente su proprietà percettibili dell'ambiente. Riferendoci all'esempio proposto, il metodo gestirà le azioni associate a *doNothing* e *inc*, definite a livello di agente, nel file .asl.

```
@Override
public boolean executeAction(String name, Structure act) {
  String functor = act.getFunctor();

  if(functor.equals("doNothing")) {
    return true;
  }

  if (functor.equals("inc")) {
    model.incValueOfAgId(
        (Integer.parseInt(name.substring(4))));
  }

  return true;
}
```

Listing 1.3. Implementazione del metodo executeAction per cyclic cellular automaton

Altri metodi che possono essere utilizzati in un environment Jason sono:

- **addPercept(L):** aggiunge il literal L alla lista globale delle percezioni; tutti gli agenti percepiranno L se sono immersi nell'ambiente prima che L sia

rimosso

- **addPercept(A, L):** aggiunge il literal L alla lista delle percezioni esclusive all'agente A, il quale sarà l'unico in grado di percepirle
- **removePercept(L):** rimuove il literal L dalla lista globale delle percezioni
- **removePercept(A,L):** rimuove il literal L dalla lista delle percezioni esclusive per l'agente A
- **clearPercepts():** elimina tutte le percezioni dalla lista globale
- **clearPercepts(A):** elimina tutte le percezioni nella lista esclusiva all'agente A

Affinché sia garantita sincronizzazione nell'ambiente, ovvero un agente computi lo step successivo solamente al termine della computazione di tutti gli altri agenti per lo step corrente, è possibile estendere la classe *TimeSteppedEnvironment* fornita da Jason. Questa classe definisce un ambiente che sincronizza tutte le azioni degli agenti, attendendo un'azione per ogni agente e, al termine della ricezione di tutte le azioni, le esegue.

Estendendo TimeSteppedEnvironment, è possibile fare override del metodo *updateAgsPercept()*, che viene chiamato dopo l'esecuzione dell'azione e prima di inviare un *continue* agli agenti. Riferendoci all'esempio proposto, in questo metodo si calcola il numero dei vicini della cellula che soddisfano i criteri definiti e si creano i rispettivi percepts, aggiungendoli alla lista esclusiva di ogni agente.

```
@Override
protected void updateAgsPercept() {
  for (int i = 0; i < model.getModelSize(); i++) {
    Location l = model.getLocationFromAgId(i);
    String name = model.getNameFromAgId(i);
    int value = model.getValueFromAgId(i);

    clearPercepts(name);

    int neighbours = calculateNeighbours();
    Literal lNeighbours = ASSyntax.createLiteral(
    "neighbours_with_upper_value",
    ASSyntax.createNumber(neighbours));

    addPercept(name, lNeighbours);
    addPercept(name, ASSyntax.createLiteral("next",
     ASSyntax.createNumber(getStep())));
  }
```

```
}
```

Listing 1.4. Implementazione del metodo updateAgsPercept()

In una visione d'insieme, possiamo suddividere il nostro sistema in tre componenti principali:

- **AutomatonEnvironment:** come descritto sopra, rappresenta l'ambiente in cui gli agenti sono immersi ed estende la classe TimeSteppedEnvironment di Jason affinché i diversi step siano sincronizzati

- **AutomatonModel:** rappresenta il modello del nostro automa ed estende la classe GridWorldModel di Jason, in modo tale da implementare una griglia in cui ogni agente ha una specifica posizione

- **AutomatonView:** è possibile associare una view a un modello in modo tale da avere una rappresentazione grafica del sistema. Facendo riferimento all'esempio proposto, la classe AutomatonView estende la classe GridWorldView di Jason e utilizza il metodo drawAgent() per la rappresentazione di ogni agente.

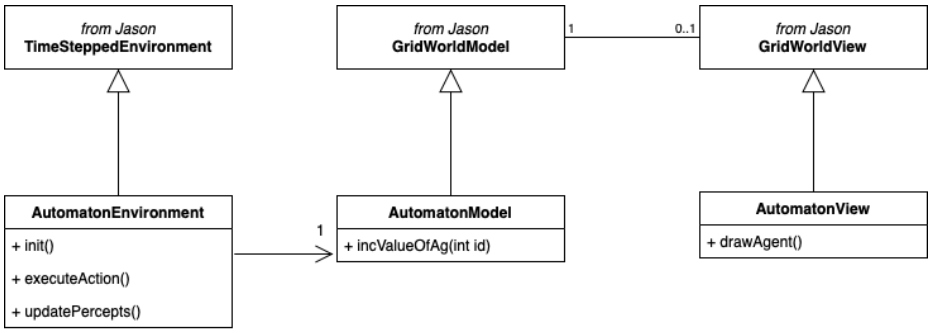

Figura 7. Diagramma delle classi per il cyclic cellular automaton in ottica Jason

Il risultato finale è mostrato nelle seguenti figure. Notiamo come a ogni agente viene associato un colore che rappresenta il suo stato. A ogni step ogni agente è autonomo e può modificare il suo stato in base a quello degli agenti vicini. Il cambiamento di stato viene evidenziato dal cambiamento di colore di quella determinata cella. Si verifica come il sistema tende a stabilizzarsi velocemente con una soglia alta; viceversa, con una soglia bassa il sistema impiegherà più tempo per stabilizzarsi.

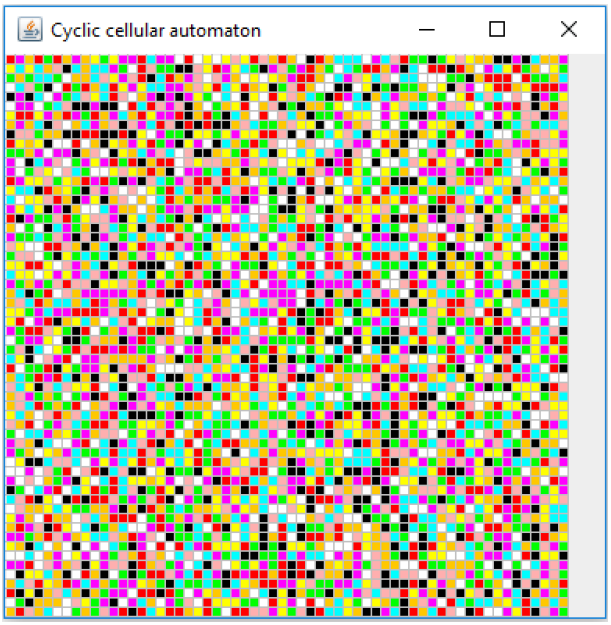

Figura 8. Configurazione iniziale random di una griglia 60x60 di agenti nel cyclic cellular automaton

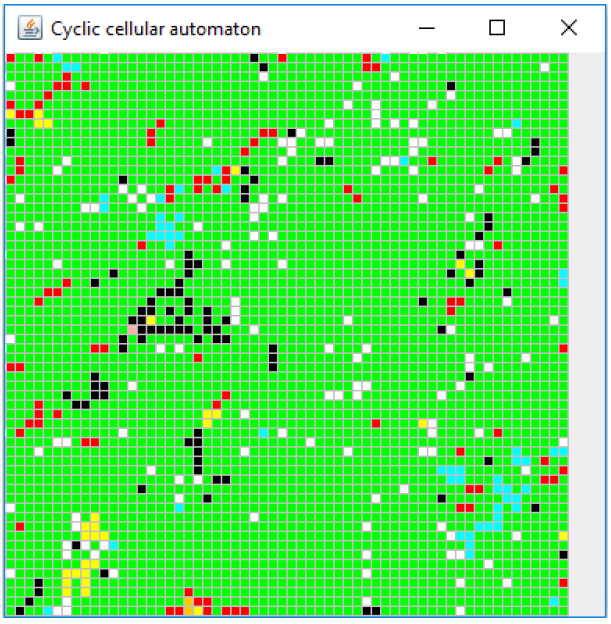

Figura 9. Stabilizzazione del cyclic cellular automaton con soglia = 1

4 Conclusioni

La SLR, analizzando in modo critico e sintetizzando i diversi documenti raccolti, è riuscita a rispondere approfonditamente alla domanda di ricerca. Al lettore viene proposta una panoramica dei principali approcci per la modellazione di sistemi biologici, con descrizione e casi di utilizzo. Poiché la disponibilità di nuovi modelli, tool o framework è all'ordine del giorno, la seguente SLR non pretende di essere completamente esaustiva, nonostante abbia incluso la letteratura rilevante relativa a tale argomento.

Riferimenti bibliografici

1. Zoltan Szallasi, Jörg Stelling, Vipul Periwal: *System Modeling in Cellular Biology - From concepts to nuts and bolts*, 2006

2. Edda Klipp, Wolfram Liebermeister, Christoph Wierling, Axel Kowald: *Systems Biology: A Textbook*, 2016

3. C.V. Suresh Babu, Eun Joo Song, Young Sook Yoo: *Modeling and simulation in signal transduction pathways: a systems biology approach*, 2005

4. Emanuela Merelli Giuliano Armano Nicola Cannata Flavio Corradini Mark d'Inverno Andreas Doms Phillip Lord Andrew Martin Luciano Milanesi Steffen Möller Michael Schroeder Michael Luck: *Agents in bioinformatics, computational and systems biology*, 2006

5. L. Milanesi, P. Romano, G. Castellani, D. Remondini, P. Liò: *Trends in modeling Biomedical Complex Systems*, 2009

6. Hong Li, Yang Cao, Linda R. Petzold, Daniel T. Gillespie: *Algorithms and Software for Stochastic Simulation of Biochemical Reacting Systems*, 2012

7. Rui-Sheng Wang, Assieh Saadatpour, Réka Albert: *Boolean modeling in systems biology: an overview of methodology and applications*, 2012

8. Daniel Machado, Rafael S Costa, Miguel Rocha, Eugénio C Ferreira, Bruce Tidor, Isabel Rocha: *Modeling formalisms in Systems Biology*, 2009

9. Steven Eschrich, Hongling Zhang, Haiyan Zhao, David Boulware, Ji-Hyun Lee, Gregory Bloom, Javier F. Torres-Roca: *Systems biology modeling of the radiation sensitivity network - a biomarker discovery platform*, 2011

10. Réka Albert: *Network Inference, Analysis, and Modeling in Systems Biology*, 2007

11. Alan Aderem: *Systems Biology: Its Practice and Challenges*, 2005

12. Choujun ZhanEmail, Lam F Yeung: *Parameter estimation in systems biology models using spline approximation*, 2011

13. Yong Wang, Xiang-Sun Zhang, Luonan Chen: *Modelling biological systems from molecules to dynamical networks*, 2012

14. Rachael Hageman Blair, David L. Trichler, Daniel P. Gaille: *Mathematical and statistical modeling in cancer systems biology*, 2012

15. E. Bartocci, P. Liò: *Computational modeling, formal analysis and tools for systems biology*, 2016

16. Marco Scutari, *Bayesian Network Modelling with Examples in Genetics and Systems Biology*, 2016

17. J.W. Pinney, D.R. Westhead, G.A. McConkey: *Petri Net representations in systems biology*, 2003

18. Di Zhao: *Differential Equation Models for Systems Biology: A Survey*, 2010

19. Susanne Ditlevsen, Adeline Samson: *Introduction to Stochastic Models in Biology*, 2013

20. M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano: *The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models*, 2003

21. PhysioDesigner, *Physiological Hierarchy Markup Language (PHML) - Language Specification*, 2011

22. Ravie Chandren Muniyandi, Abdullah Mohd. Zin: *Membrane Computing as the Paradigm for Modeling Systems Biology*, 2013

23. Safee Ullah Chaudhary, Sung-Young Shin, Daewon Lee, Je-Hoon Song Kwang-Hyun Cho: *ELECANS – an integrated model development environment for multiscale cancer systems biology*, 2013

24. R. Bardini, G. Politano, A. Benso, S. Di Carlo: *Multi-level and hybrid modelling approaches for systems biology*, 2017

25. Ozan Kahramanogullari, Luca Cardelli, Emmanuelle Caron: *An Intuitive Automated Modelling Interface for Systems Biology*, 2009

26. Vashti Galpin, Jane Hillston, Luca Bortolussi: *HYPE Applied to the Modelling of Hybrid Biological Systems*, 2008

27. Qianqian Wu, Kate Smith-Miles, Tianhai Tian: *Approximate Bayesian computation schemes for parameter inference of discrete stochastic models using simulated likelihood density*, 2014

28. Santo Motta, Francesco Pappalardo: *Mathematical modeling of biological systems*, 2012

29. Roberto Barbutia, Francesca Levia, Paolo Milazzoa, Guido Scatenab: *Probabilistic model checking of biological systems with uncertain kinetic rates*, 2012

30. M. L. Guerriero, D. Prandi, C. Priami, P. Quaglia, *Process Calculi Abstractions for Biology*, 2006

31. Cosimo Lanevea, Fabien Tarissanb: *A Simple Calculus for Proteins and Cells*, 2007

32. Steven S. Andrews, Tuan Dinh, Adam P. Arkin: *Stochastic Models of Biological Processes*, 2009

33. Federica Ciocchetta, Jane Hillston: *Process algebras in systems biology*, 2008

34. Faraz Hussain, Christopher J Langmead, Qi Mi, Joyeeta Dutta-Moscato, Yoram Vodovotz, Sumit K Jha: *Automated parameter estimation for biological models using Bayesian statistical model checking*, 2015

35. Sebnem Bora, Sevcan Emek: *Agent-Based Modeling and Simulation of Biological Systems*, 2018

36. John Wiley, Sons Ltd: *Programming Multi-Agent Systems in AgentSpeak Using Jason*, 2007

37. R. H. Bordini, J. F. Hubner, M. Wooldridge: *Programming Multi-Agent Systems in AgentSpeak Using Jason*, 2007