The outcome of the proposed summer research is expected to help the researchers determine the best route for their research. The results will give the researchers an idea of what to expect when working in vivo. Part of my responsibility as well is to work closely with my fellow peers and overseers to achieve the best results. I will be meeting with Dr. Nakano weekly, and will occasionally have group meetings with the other members of the group.

4. Project Timeline

Table 1 shows the Mr. Galal's project milestones of 2005 summer months. Dr. Nakano will be leading the proposed project. Mr. Galal will meet Dr. Nakano weekly to review Mr. Galal's progress and discuss the next step. There will be occasional group meetings and other research members (undergraduate students, and Ph. D. students in the same research group) will be invited.

Table 1: Project Time Table

Week 1	Finish creating a working stochastic model to compare with a deterministic one. Keep all results well documented.
Week 2	Identify the strengths and weaknesses of a stochastic simulation versus a deterministic simulation; and determine which aspects of the simulation should be run stochastically and which should be run deterministically
Week 3	Make detailed simulation plans. Decide what performance measures to obtain through simulations. Create a detailed document with the simulation plans.
Week 4	Continue with simulation implementation and work out many of the bugs in the simulation
Weeks 5 and 6	Conduct extensive sets of simulations according to simulation plans made in the fall quarter, week 9 and 10 (including changing variables and experimental factors to observe the effect it has on calcium and IP_3 propagation.
Week 7	Analyze simulation results from weeks 2 to 6. For analysis, visualize emerging networks formed. Find out any correlation among the network conditions and network formation
Weeks 8 and 9	Organize simulation results, and start writing a project summary. Create presentation slides about this project and be prepared for a

	meeting scheduled at the final week.
Week 10	Provide a seminar and present research results to all members.

5. References

[1] T. Hofer, Laurent Venance, and Christian Giaume. Control and Plasticity of Intercellular Calcium Waves in Astrocytes: A Modeling Approach. The Journal of Neuroscience, June 15, 2002, 22(12):4850–4859.

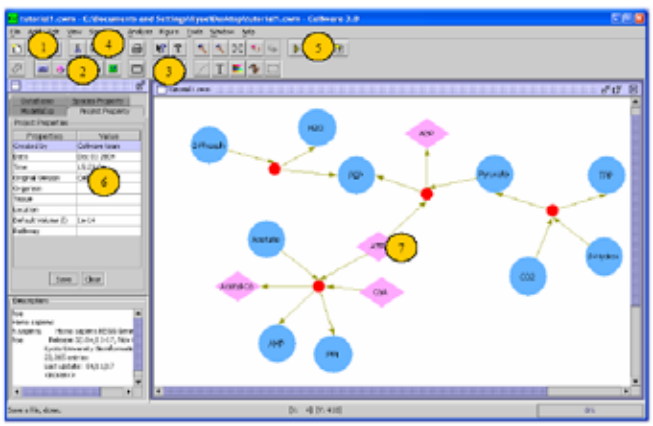
[2] Cellware Manual. Systems Biology Group Bioinformatics Institute. http://www.bii.a-star.edu.sg/

[3] D.T. Gillespie. Exact Stochastic Simulation of Coupled Chemical Reactions. The Journal of Physical Chemistry, 81(25):2340-2361, 1977.

[4] M.A. Gibson and J. Bruck. Efficient Exact Stochastic Simulation of Chemical Systems with Many Species and Many Channels. The Journal of Physical Chemistry, A 104, 1876 (2000).

[5] D.T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. The Journal of Physical Chemistry, July 22, 2001 volume 115, number4.

[6] T.S. Shimizu and D. Bray. Computational Cell Biology-The Stochastic Approach. MIT Press.


1
2
3
4
5
6
7

To create a model in Cellware, there are several means to do it. User can load ready-made models from pathway databases or SBML files (will touch them in next chapters) or manually create a new model from scratch. Hereby, we'll describe the manually creating of a model.

Creating a project – Use the New Project button  to open a new project workspace. Click on the button to create a new project. A new modelling canvas will be automatically generated.

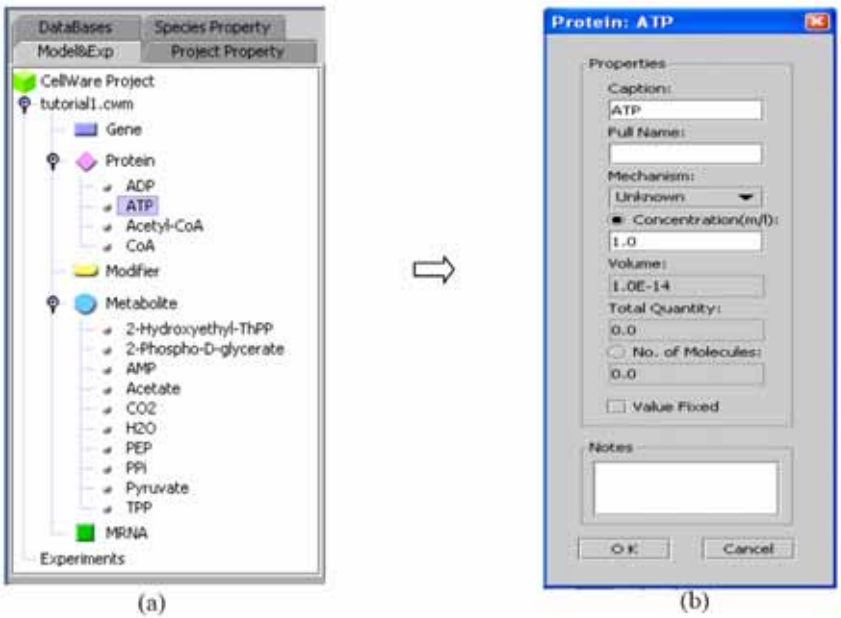
Creating a species – Use the Species buttons  to create new species in the modelling canvas. Click on the Tool once and then click on the modelling canvas where you want to place the new species.

Creating a reaction node – Use the Reaction button  to create a new reaction in the modelling canvas. Click on the button once and click in the modelling canvas where you want the new reaction created.

Creating a reaction – Use the Linking Tool  to links the species to the reaction node. Click on the tool once and click on the species you want to link with the reaction node, hold the left mouse button and drag the arrow to the reaction node.

Creating a compartment – Use the Compartment button  to create a new compartment in the modelling canvas. Click on the workspace and drag the mouse to the desired region which covers all the reactions taking place in the same compartment.

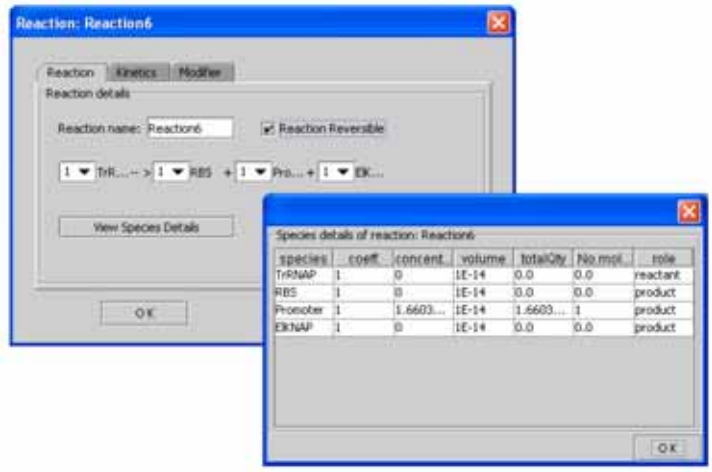
You can easily explore all the species in the model tree viewer by its category and edit any of them by double clicking on the name (Figure 3-2a, b).



(a) (b)

Defining a Reaction

- Double click on the reaction node in the modeling canvas and the *Reaction Editor* will appear. Modify the reaction stoichiometry in this panel and specify if the reaction is reversible. You may also click on the “kinetics” tab to access and modify the reaction kinetics parameters.



Reaction Kinetics

Reaction Kinetics

- The **mass action** rate law describes the rate of a typical elementary reaction where no intermediates are formed.
  - With a general equation of the form:
  - The rate determined by the mass action rate law is:

$$aA + bB \rightarrow cC$$

$$v = f([A],[B],[C],T,t) = k[A]^{\alpha}[B]^{\beta}$$

- The **Henri-Michaelis-Menten** rate law models the mechanism of enzyme catalyzed reactions. In the case of a single reactant R and a single product P:

$$E + R \leftrightarrows ER$$

$$ER \rightarrow E + P \quad \overset{k_2}{\underset{k_{-1}}{\rightleftarrows}}$$

Were E stands for the enzyme and ER the enzyme-reactant complex. And the reaction rate is determined by:

$$v = \frac{v_{max}[R]}{K_m + [R]} \qquad \mathbf{Km} = \frac{\mathbf{k_{-1}}}{\mathbf{k_1}} \qquad \mathbf{Vmax = k_2[E_t]}$$

- If the formation of ER complex k-1 is faster than the dissociation of the ER complex k1, then Km is small. The smaller the Km, the tighter the ER complex.

- **Competitive Inhibition** considers a third reaction to the Michelis-Menten reactions:

$$E + I \leftrightarrows EI$$

I is the inhibitor and EI is a complex of the catalyst and the inhibitor. The rate law then becomes:

$$v = \frac{v_{max}[R]}{[R] + K_m\left(1 + \frac{[I]}{K_I}\right)}$$

- Kf is the dissociation constanst of the complex EI

- In **non-competitive inhibition** it is assumed that the inhibitor can bind to the enzyme-reactant complex as well as to the free enzyme preventing the enzyme-reactant to break and form the product:

$$\begin{aligned} E + R &\rightleftharpoons ER \\ ER &\rightarrow E + P \\ E + I &\rightleftharpoons EI \\ ER + I &\rightleftharpoons ERI \\ EI + R &\rightleftharpoons ERI. \end{aligned}$$

The modified rate law becomes:

$$v = \frac{v_{max}[R]}{[R]\left(1 + \frac{[I]}{K_I}\right) + K_m\left(1 + \frac{[I]}{K_I}\right)}$$

- **Hill's Cooperativity**: cooperative binding happens when the affinity of the ligand for a macromolecule depends on the amount of ligand already bound. The cooperative binding can be either positive or negative, indicating that the affinity is either increased or decreased by the binding of the ligand.

  The rate law is:

$$v = \frac{v_{max}[\mathrm{R}]}{[\mathrm{R}]\left(1 + \frac{[\mathrm{I}]}{K_I}\right) + K_m\left(1 + \frac{[\mathrm{I}]}{K_I}\right)}$$

- **User Defined**: The reaction rate formula can be given as an arbitrary function of the reaction component concentrations. ***No numbers in formula, only variables.***

Be sure to hit enter after entering value

Creating Model (SBML)

- Cellware can import SBML files (File>Import from XML files), and can export XML files (File>Export to XML files) in the following formats:
  - SBML level 1 version 1
  - SBML level 1 version 2
  - SBML level 2 version 1 (seems to have an error though)

- More at end of slides

Simulation

- **Deterministic Simulation**: algorithms solve a group of non-linear differential equations assuming that no noise or any stochastic variation are present in the reaction.
  - **Euler forward method**:
    - Euler forward method is the fastest solver for most non-stiff ODE problems, has lower order of accuracy and small time step size.
  - **Euler backward method**:
    - Almost the fastest solver for most stiff ODE problems, first order accuracy.
  - **Trapezoidal method**:
    - Fastest solver for stiff ODE problems, second order accuracy.
  - **Explicit 4th order Runge-Kutta method**:
    - One-step solver for non-stiff ODE problem, high order accuracy and large time step size.
  - **Rosenbrock method**:
    - A special case of the generalized Runge-Kutta method, high order accuracy and large time step size.
  - **Advanced ODE solver**:
    - It is an automated explicit Adams-Bashforth four step method, it can automatically detect whether the problem is stiff or non-stiff and use small time step size.

- **Stochastic algorithms**: takes into consideration stochastic or random events, does not assume concentration changes continuously over time (which is what the deterministic simulation assumes).
  - **Gillespie's Direct method**:
    - An exact stochastic simulation algorithm for homogeneous, well-mixed chemical reaction systems. Requires substantial amount of computational effort to simulate a complex system.
  - **Gibson Next Reaction method**:
    - Omits the repetitive activities in Gillespie's direct method, it is much faster than the Gillespie's direct method, especially for large scale problems.
  - **Explicit Tau-Leap method**:
    - Uses a larger time step than Gillespie's direct algorithm, it is much faster than Gillespie's direct algorithm, but the accuracy is highly sensitive to the error control mechanisms and the tolerance.

- **Hybrid Algorithms**: Combines different numerical algorithms to simulate a biological system, the idea is that the advantages of one algorithm will overcome the disadvantages of another.
  - StochODE method:
    - Integrates differential equations with stochasticity by introducing an external noise to the ODE, which is generated based on Poisson distribution of the quantity of the species. Fast in large scale systems, but accuracy is sacrificed.

Important Notice

Stochastic and Hybrid algorithms are only available for the ***mass action*** rate law.

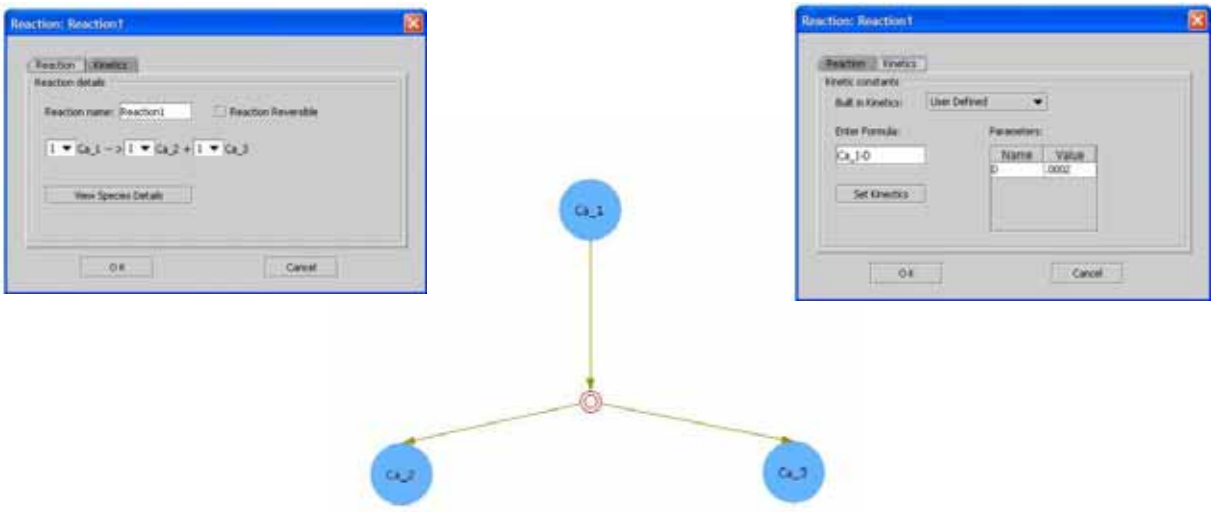
- **Deterministic** algorithms require the following simulation parameters:
  - ***Integrator***: This indicates the ODE integrator scheme – Euler forward, Runga Kutta 4th Order or Advanced ODE solver.
  - ***Total Time***: This is the duration of simulation in seconds.
  - ***Save Period***: This is the time interval at which the results would be plotted or saved to file.
  - ***Tolerance***: Tolerance limitation for the ODE solvers.

- **Hybrid** algorithm requires the same input parameters as the deterministic solvers.

- **Stochastic** algorithms require the following simulation parameters
  - *Method*: The user has an option of choosing from one of the three stochastic simulation algorithms – Gillespie Direct, Gibson-Gillespie and Tau Leap.
  - *Tolerance*: It species a termination criteria for the algorithm such that if the sum of propensities of all reaction fall below the *tolerance* the simulation would be terminated. The default value is set to 1e-6. Essentially it detects whether or not there are any feasible reactions during the simulation iterations.
  - *Number of Experiments*: This is the number of times a simulation would be repeated. This is required for stochastic simulation because the time series is different for different runs and to get a statistically relevant result multiple time series should be averaged. By default this parameter is 1. Note that multiple runs can only be done in the batch mode of Cellware.
  - *Number of iteration*: This is another termination criteria which species the maximum number of iterations of the simulation algorithm. The default value is 1000000.
  - *Total Time*: This is the duration of simulation in seconds.
  - *Save Period*: This is the time interval at which the results would be plotted or saved to file.

Creating a Model (SBML Level 1, Version 2)

GUI for Simple Ca^{2+} Model



Ca2+ Model (SBML equavalent)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<sbml level="1" version="2" xmlns="http://www.sbml.org/sbml/level1">
    <model name="Ca test.cwm">
        <listOfCompartments>
            <compartment name="DefaultCompartment" volume="1.0E-14" />
        </listOfCompartments>
        <listOfSpecies>
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="1.0E-7" name="Ca_1" />
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="0.0" name="Ca_2" />
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="0.0" name="Ca_3" />
        </listOfSpecies>
        <listOfReactions>
            <reaction name="Reaction1" reversible="false">
                <listOfReactants>
                    <speciesReference species="Ca_1" stoichiometry="2" />
                </listOfReactants>
                <listOfProducts>
                    <speciesReference species="Ca_2" stoichiometry="1" />
                    <speciesReference species="Ca_3" stoichiometry="1" />
                </listOfProducts>
                <kineticLaw formula="(Ca_1-D)">
                    <listOfParameters>
                        <parameter name="D" value="2.0E-4" />
                    </listOfParameters>
                </kineticLaw>
            </reaction>
        </listOfReactions>
    </model>
</sbml>
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<sbml level="1" version="2" xmlns="http://www.sbml.org/sbml/level1">
    <model name="Ca test.cwm">
        <listOfCompartments>
            <compartment name="DefaultCompartment" volume="1.0E-14" />
        </listOfCompartments>
        <listOfSpecies>
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="1.0E-7" name="Ca_1" />
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="0.0" name="Ca_2" />
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="0.0" name="Ca_3" />
        </listOfSpecies>
        <listOfReactions>
            <reaction name="Reaction1" reversible="false">
                <listOfReactants>
                    <speciesReference species="Ca_1" stoichiometry="2" />
                </listOfReactants>
                <listOfProducts>
                    <speciesReference species="Ca_2" stoichiometry="1" />
                    <speciesReference species="Ca_3" stoichiometry="1" />
                </listOfProducts>
                <kineticLaw formula="(Ca_1-D)">
                    <listOfParameters>
                        <parameter name="D" value="2.0E-4" />
                    </listOfParameters>
                </kineticLaw>
            </reaction>
        </listOfReactions>
    </model>
</sbml>
```

Specifies model name


```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<sbml level="1" version="2" xmlns="http://www.sbml.org/sbml/level1">
    <model name="Ca test.cwm">
        <listOfCompartments>
            <compartment name="DefaultCompartment" volume="1.0E-14" />
        </listOfCompartments>
        <listOfSpecies>
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="1.0E-7" name="Ca_1" />
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="0.0" name="Ca_2" />
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="0.0" name="Ca_3" />
        </listOfSpecies>
        <listOfReactions>
            <reaction name="Reaction1" reversible="false">
                <listOfReactants>
                    <speciesReference species="Ca_1" stoichiometry="2" />
                </listOfReactants>
                <listOfProducts>
                    <speciesReference species="Ca_2" stoichiometry="1" />
                    <speciesReference species="Ca_3" stoichiometry="1" />
                </listOfProducts>
                <kineticLaw formula="(Ca_1-D)">
                    <listOfParameters>
                        <parameter name="D" value="2.0E-4" />
                    </listOfParameters>
                </kineticLaw>
            </reaction>
        </listOfReactions>
    </model>
</sbml>
```

Specifies compartments

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<sbml level="1" version="2" xmlns="http://www.sbml.org/sbml/level1">
    <model name="Ca test.cwm">
        <listOfCompartments>
            <compartment name="DefaultCompartment" volume="1.0E-14" />
        </listOfCompartments>
        <listOfSpecies>
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="1.0E-7" name="Ca_1" />
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="0.0" name="Ca_2" />
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="0.0" name="Ca_3" />
        </listOfSpecies>
        <listOfReactions>
            <reaction name="Reaction1" reversible="false">
                <listOfReactants>
                    <speciesReference species="Ca_1" stoichiometry="2" />
                </listOfReactants>
                <listOfProducts>
                    <speciesReference species="Ca_2" stoichiometry="1" />
                    <speciesReference species="Ca_3" stoichiometry="1" />
                </listOfProducts>
                <kineticLaw formula="(Ca_1-D)">
                    <listOfParameters>
                        <parameter name="D" value="2.0E-4" />
                    </listOfParameters>
                </kineticLaw>
            </reaction>
        </listOfReactions>
    </model>
</sbml>
```

Specifies species


```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<sbml level="1" version="2" xmlns="http://www.sbml.org/sbml/level1">
    <model name="Ca test.cwm">
        <listOfCompartments>
            <compartment name="DefaultCompartment" volume="1.0E-14" />
        </listOfCompartments>
        <listOfSpecies>
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="1.0E-7" name="Ca_1" />
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="0.0" name="Ca_2" />
            <species boundaryCondition="false" compartment="DefaultCompartment" initialAmount="0.0" name="Ca_3" />
        </listOfSpecies>
        <listOfReactions>
            <reaction name="Reaction1" reversible="false">
                <listOfReactants>
                    <speciesReference species="Ca_1" stoichiometry="2" />
                </listOfReactants>
                <listOfProducts>
                    <speciesReference species="Ca_2" stoichiometry="1" />
                    <speciesReference species="Ca_3" stoichiometry="1" />
                </listOfProducts>
                <kineticLaw formula="(Ca_1-D)">
                    <listOfParameters>
                        <parameter name="D" value="2.0E-4" />
                    </listOfParameters>
                </kineticLaw>
            </reaction>
        </listOfReactions>
    </model>
</sbml>
```

Specifies reaction: reactants, products, kinetic rate law, parameters (for rate law)

Stochastic Algorithms

Mohamed Galal

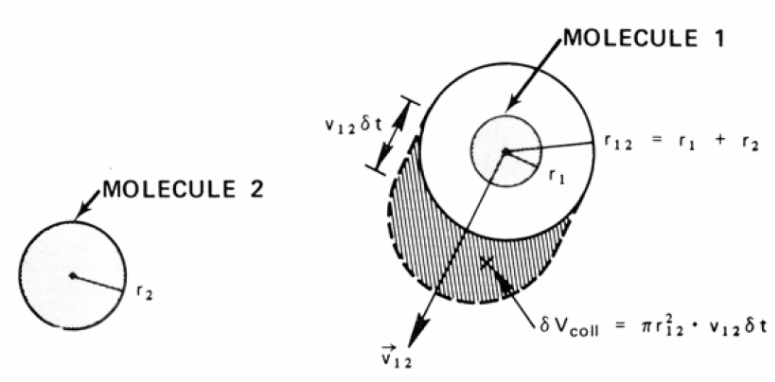
03/29/2005

Stochastic Summary

- Idea is to represent individual molecules rather than concentrations of molecular species.
  - Many crucial events in living cells depend on the interaction of small numbers of molecules, and hence are sensitive to the stochasticity of the reaction.
  - Under these conditions, the deterministic approach fails to predict the behavior of the system accurately.
- The algorithms:
  - Gillespie's stochastic simulation algorithm
  - Gillespie's first reaction method
  - Gillespie's tau-leap method
  - StochSim
  - Gibson's next reaction method

Gillespie's Stochastic Simulation Algorithm

- Consider first two molecules, S_1 and S_2, there will be a collision when the center-to-center distance between the two decreases to $r_{12}=r_1 + r_2$ (r being the radius).
- To calculate the rate, consider the following:
  - v_{12}, the speed of molecule 1 relative to molecule 2. Over a time interval dt, molecule 1 will move a volume $dVcoli = \pi r_{12}^2 v_{12} dt$. If molecule 2 lies inside dVcoli at time t, then the two molecules will colide in the time interval (t, t+dt).



The Journal of Pysical Chemistry, Vol. 81, No. 25, 1977

- The next step is to estimate the number of S_2 molecules whose centers are inside dVcoli, divide that by dt, and lake the limit as $dt \rightarrow 0$. However, as $dVcoli \rightarrow 0$, the number of S_2 molecules that are inside dVcoli will be either 0 or 1.
- But since the system is in thermal equilibrium, the molecules are distributed randomly and uniformily. Therefore the probability that S_2 is found inside dVcoli is:
  - $\overline{dVcoli/V} = V^{-1} \pi r_{12}^2 \overline{v}_{12} dt$ = "average probability that a particular 1-2 molecular pair will collide in the next vanishingly small time interval dt"
  - $\overline{v}_{12} = (8kT/\pi m_{12})^{1/2}$, and m_{12} is the reduced mass.
- There are X_1X_2 distinct 1-2 molecular pairs, then:
  - $X_1X_2 V^{-1} \pi r_{12}^2 v_{12} dt$ = "probability that a 1-2 collision will occur somewhere inside V in the next infintesimal time interval (t, t + dt)
- In general it can be taken as:
  - $X_1X_2 c_1 dt$ = "probability that an R_1 reaction will occur somewhere inside V in the next infintesimal time interval (t, t + dt)"
    - Were c_1 depends only on the physical properties of the two molecules and the temperature of the system.

- In general, for a system of X_i molecules of chemical species S_i (i = 1,…,N), and the N species can react through M reaction channels R_μ (μ = 1,…,M).
  - Then there are M constants c_μ (μ = 1,…,M) which depend on physical properties of the molecules and the temperature of the system
  - c_μ dt = "average probability that a particular combination of R_μ reactant molecules will react accordingly in the next infinitesimal time interval dt"
  - This is the definition of c_μ, and the fundamental hypothesis of the stochastic formulation of chemical kinetics.

Stochastic Simulation Algorithm

- It is not derived from the Master equation (which gives an almost exact answer, but it is difficult to find the solution even with a computer).
- The algorithm asks two questions:
  - When will the next reacton occur?
  - What kind of reaction will it be?
- $P(\tau, \mu)$ dt = "probability that, given the sate $(X_1,\ldots,X_N)$ at time t, the next reaction in V will occur in the infinitesimal time interval $(t + \tau, t + \tau + d\tau)$, and will be an R_μ reaction.
  - $P(\tau, \mu)$ is called the "reaction probability density function"
  - τ is the next reaction time, μ the reaction type
- The first way of finding values to τ and μ, is to derive from the fundamental hypothesis an expression for $P(\tau, \mu)$.

- First, define for each reaction R_μ a function h_μ:
  - h_μ = "number of distinct R_μ molecular reactant combinations available in the state (X_1, X_2,…, X_N) (μ = 1,…,M)"
  - If the reaction is $S_1 + S_2 \rightarrow$ anything, then $h_\mu = X_1 X_2$.
- Therefore: $a_\mu\, dt = h_\mu c_\mu$ = "probability that an R_μ reaction will occur in V in (t, t + dt), given that the system is in the state (X_1,…X_N) at time t (μ = 1,…,M)"
- $P(\tau, \mu)\, dt = P_o(\tau) \times a_\mu\, dt$
  - $P_o(\tau)$ = the probability that at time t in state (X_1,…X_N) that no reaction will occur in (t, t + τ).
  - $a_\mu dt$ = action R_μ will occur in V in (t + τ, t + τ + d τ).

$$P_0(\tau) = \exp\left[-\sum_{\nu=1}^{M} a_\nu \tau\right]$$

$$P(\tau, \mu) = \begin{cases} a_\mu \exp(-a_0\tau) & \text{if } 0 \leqslant \tau < \infty \text{ and} \\ & \mu = 1,\ldots,M \\ 0 \quad \text{otherwise} \end{cases}$$

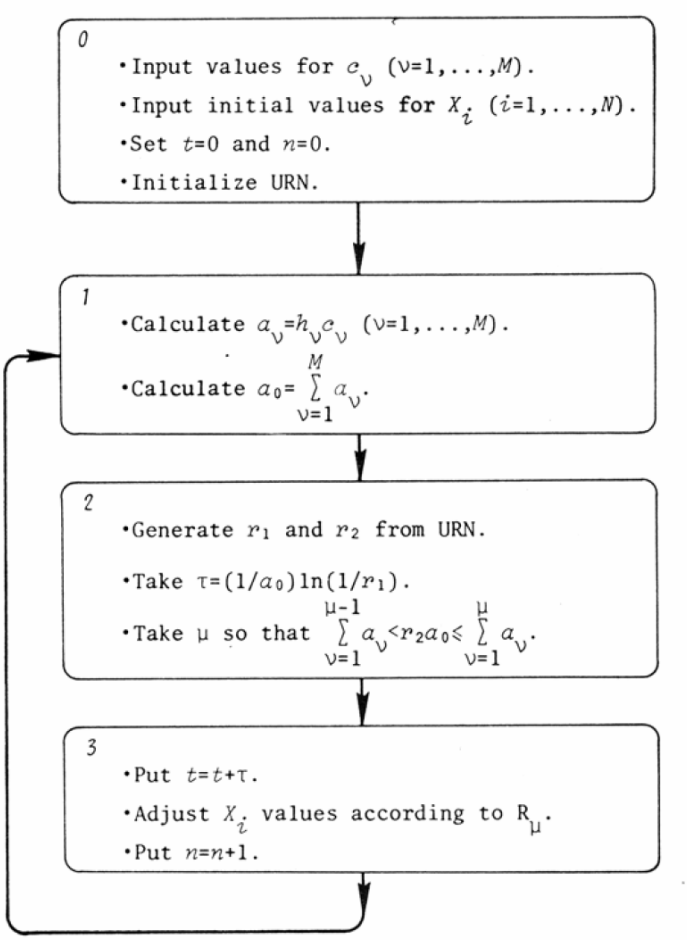
$$a_0 \equiv \sum_{\nu=1}^{M} a_\nu \equiv \sum_{\nu=1}^{M} h_\nu c_\nu$$

- $P(\tau, \mu)$ depends on all the reaction constants and on the current numbers of molecles of all reactant species.
- What is now needed is a way to generate a pair of (τ, μ) from the set of random pairs whose probability density functions is $P(\tau, \mu)$.
  - This can be done using an "unit-interval uniform random number generator" (URN).
    - The URN generates a number r, such that the probability that any generated r will fall inside any given subinterval [a,b] of the unit interval [0,1] is equal to b-a.
- With two generated numbers, r_1 and r_2, from the URN:

$$\tau = (1/a_0) \ln (1/r_1)$$

$$\sum_{\nu=1}^{\mu-1} a_\nu < r_2 a_0 \leqslant \sum_{\nu=1}^{\mu} a_\nu$$

- To compute the last equation, it's coded as:
  - Add the successive values of $a_1, a_2, \ldots a_v$ in a do-loop until there sum is equal to or greater than $r_2 a_o$; then take the the last index of a_v to equal μ.
- The algorithim is summarized as:



- In going from step 3 to step 1, you only need to recalculate the quantities a_μ corresponding to reactions R_μ whose reactant population levels were just altered.
- a_o can be recalculated by adding a_o to the difference between each newly changed a_v and its old value.

- This algorithm depends on the number of species N, not on the number of molecules.
- On the other hand, a stochastic simulation often requires a lot of time to execute.
  - This limits the total number of molecules involved.

Summary of Algorithm

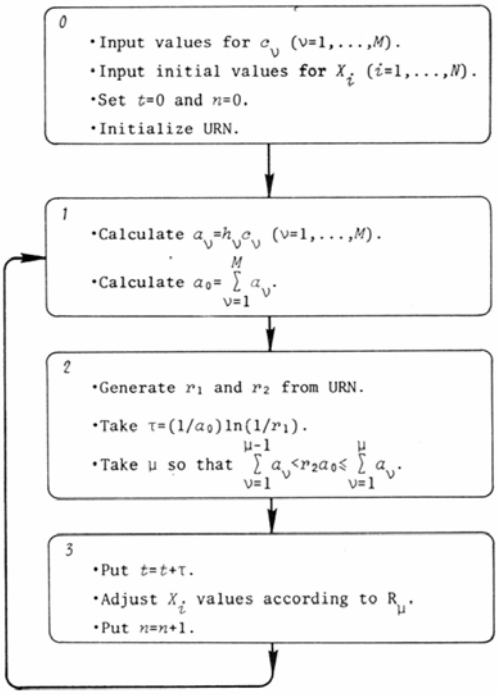
- $P(\tau, \mu)\, dt = P_o(\tau) \times a_\mu\, dt$
  - $P_o(\tau)$ = the probability that at time t in state $(X_1,\ldots X_N)$ that no reaction will occur in $(t, t + \tau)$.
  - $a_\mu\, dt$ = subsequent probability that reaction R_μ will occur in V in $(t + \tau, t + \tau + d\tau)$.

$$P(\tau, \mu) = \begin{cases} a_\mu \exp(-a_0\tau) & \text{if } 0 \leqslant \tau < \infty \text{ and} \\ & \mu = 1,\ldots,M \\ 0 \quad \text{otherwise} & \end{cases}$$

$$P_0(\tau) = \exp[-\sum_{\nu=1}^{M} a_\nu \tau] \qquad a_0 \equiv \sum_{\nu=1}^{M} a_\nu \equiv \sum_{\nu=1}^{M} h_\nu c_\nu$$

- Generate two random numbers r_1 and r_2 to generate τ and μ.

$$\tau = (1/a_0) \ln (1/r_1) \qquad \sum_{\nu=1}^{\mu-1} a_\nu < r_2 a_0 \leqslant \sum_{\nu=1}^{\mu} a_\nu$$



Gillespie's First Reaction Method

- Previous algorithim generates μ and τ directly.
- Generates a putative time τ_i for each reaction to occur
  - time that the reaction would occur if no other reaction occurred first
- Sets μ to be the reaction whose putative time is first, and least τ to be the putative time τ_μ.
  - Initialize (set initial numbers of molecules, set t=0).
  - Calculate the propensity function, a_i, for all i.
  - For each i, generate a putative time, τ_i, according to an exponential distribution with pararmeter a_i.
  - Let μ be the reaction whose putative time, τ_μ, is least.
  - Let τ be τ_μ.
  - Change the number of molecules to reflect execution of reaction μ. Set t to $t \leftarrow \tau$.
  - Go to Step 2.
- Algorithim uses same distributions as in the Direct Method.
  - Algorithim uses r random numbers per iteration (where r is the number of reactions), takes time proportional to r to update a_i's, and takes time proportional to r to identify the smallest τ_μ.

τ -Leap Method

τ -Leap Method

- Instead of moving one reaction at a time, skip over several reactions in a way that would not cause a significant change in values.
- For a system with S_i molecules and R_j reactions, the propensity function of Rj is specified as:

$$a_j(\mathbf{x})dt \equiv \text{the probability, given } \mathbf{X}(t)=\mathbf{x}, \text{ that one } R_j \text{ reaction will occur somewhere inside } \Omega \text{ in the next infinitesimal time interval } [t,t+dt) \quad (j=1,\ldots,M).$$

- We specify τ to be:
  - Small enough that the change in the state during [t,t + τ) will be so small that no propensity function will suffer significant (macroscopically non-infinitesimal) change in value.
  - Assuming this is satisfied, then each propensity function for each channel R_j will remain virtually constant at the value $a_j(x)$.
  - That means that $a_j(x)dt$ will give the probability that Rj will fire during any infinitesimal interval dt inside [t,t + τ), regardless of what the other channels are doing.

- Consider the probability function Q (the joint probability density function):

$$Q(k_1,\ldots,k_M|\tau;\mathbf{x},t)$$
$$\equiv \text{the probability, given } \mathbf{X}(t)=\mathbf{x}, \text{ that in the time interval } [t,t+\tau) \text{ exactly } k_j \text{ firings of reaction channel } R_j \text{ will occur, for each } j=1,\ldots,M.$$

- K is specified as:

$$K_j(\tau;\mathbf{x},t) \equiv \text{the number of times, given } \mathbf{X}(t)=\mathbf{x}, \text{ that reaction channel } R_j \text{ will fire in the time interval } [t,t+\tau) \ (j=1,\ldots,M).$$

- K can be computed using a Poisson random variable: $K_j(\tau;\mathbf{x},t)=\mathcal{P}(a_j(\mathbf{x}),\tau) \quad (j=1,\ldots,M).$
  - Since the K's are independent, Q can simply be the product of the density functions of the individual Poisson random variables:

$$Q(k_1,\ldots,k_M|\tau;\mathbf{x},t)=\prod_{j=1}^{M} P_{\mathcal{P}}(k_j;a_j(\mathbf{x}),\tau)$$

- The procedure is:
  - Generate for each reaction channel Rj a sample value kj of the Poisson random variable
    - kj will be the number of times reaction channel Rj fires in [t,t + τ).
  - Since each firing of Rj changes Si, the net change in the system in [t,t + τ) is:

$$\boldsymbol{\lambda}=\sum_{j=1}^{M} k_j \boldsymbol{\nu}_j$$

  - Were

$$\nu_{ji} \equiv \text{the change in the number of } S_i \text{ molecules produced by one } R_j \text{ reaction } (j=1,\ldots,M;i=1\ldots,N),$$

  - Increment t by t + τ and x by x + λ .
- If the reactant populations are very large, it will take a very large number of reactions to change the propensity functions significantly. Therefore we can satisfy the Leap condition with a choice for τ that allows many reactions events to occur in [t,t + τ).
- If τ is $1/a_o(x)$ or smaller, the results produced by the leap will be that of the exact SSA (stochastic simulation algorithm).

The Langevin Method

- Assume we can leap over a large number of reactions and still satisfy the Leap Condition. And assume that the expected number of firings of each reaction in [t,t + τ) obeys:

$$\langle \mathcal{P}(a_j(\mathbf{x}),\tau)\rangle = a_j(\mathbf{x})\tau \gg 1 \,(\forall j = 1,...,M)$$

- Then the following simplification can be made:
  - Since the Poisson random variable, when at >> 1, will be approximated by a normal random variable with the same mean and variance. Then the number of firings of Rj is:

$$\begin{aligned} K_j(\tau;\mathbf{x},t) &= \mathcal{P}_j(a_j(\mathbf{x}),\tau) \\ &\approx \mathcal{N}_j(a_j(\mathbf{x})\tau, a_j(\mathbf{x})\tau) \\ K_j(\tau;\mathbf{x},t) &= a_j(\mathbf{x})\tau + (a_j(\mathbf{x})\tau)^{1/2}\mathcal{N}_j(0,1) \quad (j = 1,...,M) \end{aligned}$$

- The procedure:

Langevin Method: Suppose it is possible to choose τ so that (i) the Leap Condition is satisfied, *and* (ii) $\tau \gg \text{Max}_j\{1/a_j(\mathbf{x})\}$. Then for each $j = 1,...,M$, generate a sample value n_j of the "unit normal" random variable $\mathcal{N}(0,1)$ and put $k_j = a_j(\mathbf{x})\tau + (a_j(\mathbf{x})\tau)^{1/2} n_j$. Finally, compute $\boldsymbol{\lambda} = \Sigma_j k_j \boldsymbol{\nu}_j$, and effect the leap by replacing t by $t+\tau$ and $\mathbf{x}$ by $\mathbf{x}+\boldsymbol{\lambda}$.

Selecting τ

- Specify ε to be beween 0 and 1; then τ becomes:

$$\tau = \min_{j \in [1,M]} \left\{ \epsilon a_0(\mathbf{x}) \Big/ \left| \sum_{i=1}^{N} \xi_i(\mathbf{x}) b_{ji}(\mathbf{x}) \right| \right\}$$

- where ξ is the mean or expected state change in a unit time:

$$\boldsymbol{\xi}(\mathbf{x}) \equiv \sum_{j=1}^{M} a_j(\mathbf{x}) \boldsymbol{\nu}_j$$

- and b is:

$$b_{ji}(\mathbf{x}) \equiv \frac{\partial a_j(\mathbf{x})}{\partial x_i} \quad (j = 1,...,M; i = 1,...,N)$$

Estimated-Midpoint τ-Leap Method

Estimated-Midpoint τ-Leap Method: For the selected leaping time τ (which satisfies the Leap Condition), compute the *expected* state change $\overline{\boldsymbol{\lambda}} = \tau \Sigma_j a_j(\mathbf{x}) \boldsymbol{\nu}_j$ during $[t, t+\tau)$. Then, with $\mathbf{x}' \equiv \mathbf{x} + [\overline{\boldsymbol{\lambda}}/2]$, generate for each $j = 1,...,M$ a sample value k_j of the Poisson random variable $\mathcal{P}(a_j(\mathbf{x}'), \tau)$. Compute the actual state change, $\boldsymbol{\lambda} = \Sigma_j k_j \boldsymbol{\nu}_j$, and effect the leap by replacing t by $t+\tau$ and $\mathbf{x}$ by $\mathbf{x}+\boldsymbol{\lambda}$.

StochSim

StochSim

- Each molecule is represented as an individual object, and dummy molecules "pseudo-molecules" are also included (which represent the chance of nothing happening).
- In each time slice, StochSim chooses one molecule at random from the population of "real" molecules, and then picks from the entire population (including pseudo-molecules).
  - If two real molecules are chosen then they are tested for all possible bimolecular reactions for that particular combination.
  - If one real molecule and one pseudo-molecule is chosen, then it is tested for all possible unimolecular reactions.
- Reaction probabilities are pre-computed at initilization and stored in a table.
- Once the pair is chosen, the probabilites are retrieved from the table, then StochSim computes a "cumulative probability" for each possible outcome.
- This set is then compared with a random number to choose which reaction, if any, occurs.
- If a reaction occurs, the system is updated and then runs this again.

- The probabilites in the table are calculated on the basis of the following:
  - Deterministic rate constant
  - Size of the time increment
  - Number of molecules in the system (n)
  - Number of pseudo-molecules in the system (n_o)
  - Volume of the system (V)
- Using these, the probabilites for uni and bi-molecular rections, p_1 and p_2 respectively, are obtained:

$$p_1 = \frac{k_1 n(n+n_0)\Delta t}{n_0} \qquad p_2 = \frac{k_2 n(n+n_0)\Delta t}{2N_A V}$$

- Proof is as follows:
  - Consider the following unimolecular reaction for A: $\frac{d[A]}{dt} = -k_1[A]$
  - Then $\Delta n_A = -k_1 n_A \Delta t$
  - The expected value of $-\Delta n_A$ is taken as:

$$\begin{aligned} -\Delta n_A \quad = \quad & \Pr(\text{molecule of A is selected in the first selection}) \\ & \times \Pr(\text{pseudo-molecule is selected in the second selection}) \\ & \times p_1 \end{aligned}$$

$$-\Delta n_A \quad = \quad \frac{n_A}{n} \times \frac{n_0}{n+n_0} \times p_1 \quad \longrightarrow \quad p_1 = \frac{k_1 n(n+n_0)\Delta t}{n_0}$$

- For a bimolecular reaction with substrates B and C:

$$\frac{d[B]}{dt} = -k_2[B][C]$$

$$\Delta n_B = -\frac{k_2 n_B n_C \Delta t}{2N_A V}$$

$$\begin{aligned} -\Delta n_B \quad = \quad & \{\Pr(\text{molecule of B is selected in the first selection}) \\ & \times \Pr(\text{molecule of C is selected in the second selection}) \\ & \times p_2\} \\ & +\{\Pr(\text{molecule of C is selected in the first selection}) \\ & \times \Pr(\text{molecule of B is selected in the second selection}) \\ & \times p_2\} \end{aligned}$$

$$-\Delta n_B \quad = \quad 2 \times \frac{n_B}{n} \times \frac{n_C}{n + n_0} \times p_2$$

$$p_2 = \frac{k_2 n(n + n_0)\Delta t}{2N_A V}$$

- When a molecular species in a system can exist in more than one state, the program specifies it as a “multistate molecule” with binary flags.
  - The flags can be modified each time step or calculated instantaneoulsy (“fast flags”) according to a probability.
- The algorithm may seem complicated, but even on a slow computer, it can carry out hunderds and thousands of iterations every second.

Comparison with Gillespie

- Gillespie:
  - Makes time steps of variable length, based on reaction rate constatns and population size of each chemical species.
  - In each iteration, one random number is used to determine when the next reaction will occur, and another what the reaction is.
  - It cannot associate physical quantities with each molecule, nore trace a particular molecule.
    - Without the ability to associate positional and velocity information with each particle, the algorithm cannot easily simulate diffusion, localization or spatial heterogeneity.
  - It cannot easily handle the reactions of multistate molecules.
    - A detailed multistate model may contain ten million distinct chemical reactions.
  - Its best for spatially homogeneous, thermodynamically equilibrated systems.
- StochSim:
  - Can be slower in calculating the outcome of a small set of simple biochemical reactions, especially when the number of molecules is large.
  - If the sysem contains molecules that are in large number of states, then StochSim will be faster and more close to reality.
  - It is easy to trace a single molecule.

The Next Reaction Method

- Gillespie's First Reaction Method has three activities that occur every iteration with time proportional to r:
  - updating all r of the a_i's
  - generating a putative time, τ_i, for each i
  - identifying the smallest putative time, τ_μ.
- The Next Reaction Method does away with these:
  - store τ_i, not just a_i.
  - recalculate a_i only if it changes
    - this is done by analyzing the reactions beforeand and determining which reactions change with a_i.
  - re-use τ_i's where appropriate
  - switch from relative time (time between reactions) to absolute time
    - for reactions whose a_i has not changed, the putative time τ_i wil not have to change
  - use appropriate data structures to store a_i's (and τ_i's) so that updating those that change will be eficient.
    - use *indexed priority queue*

The Next Reaction Method

1. Initialize
   i. Set initial numbers of molcules, set t ← 0, generate a dependency graph G
   ii. Calculate the propensity function a_i, for all i
   iii. For each i, generate a putative time, τ_i, according to an exponenetial distribution with parameter a_i.
   iv. Store the τ_i values in an indexed priority queue P.
2. Let μ be the reaction whose putative time, τ_i, stored in P, is least
3. Let τ be τ_μ.
4. Change the number of molecules to reflect execution of reaction μ. Set t ← τ .
5. For each edge (μ, $\acute{\alpha}$) in the dependency graph G,
   i. Update $a_{\acute{\alpha}}$.
   ii. If $\acute{\alpha}$!= μ, set $\tau_{\acute{\alpha}} \leftarrow (a_{\acute{\alpha},old}/a_{\acute{\alpha},new})(\tau_{\acute{\alpha}} - t) + t$
   iii. If $\acute{\alpha}$ = μ, generate a randum numer, p, according to an exponential distribution with parameter a_μ, and set $\tau_{\acute{\alpha}} \leftarrow p + t$
   iv. Replace the old $\tau_{\acute{\alpha}}$ value in P with the new value
6. Go to Step 2.

- **Uses approximatly one random number per iteration**