Developing A Bitwise Macromolecular Assembly Simulator

By

Zaikun Xu

Submitted to the Department of Molecular Biosciences and the
Graduate Faculty of the University of Kansas
in partial fulfillment of the requirements for the degree of
Master of Arts

Eric Deeeds, Chairperson

Committee members

John Karanicolas

Christian Ray

Date defended: June 05, 2014

The Dissertation Committee for Zaikun Xu certifies
that this is the approved version of the following dissertation :

Developing A Bitwise Macromolecular Assembly Simulator

Eric Deeeds, Chairperson

Date approved: June 05, 2014

Abstract

Macromolecular machines play fundamental roles in many cellular tasks, from intracellular transport to protein synthesis and degradation. The majority of these machines must adopt a particular quaternary structure in order to function, and so understanding their assembly represents a critical component of our understanding of overall cellular physiology. Developing a theoretical and conceptual understanding of assembly has been hampered by the lack of general, efficient and scalable computational tools for simulating assembly processes.

In this work, we develop a new framework that employs a bitwise representation of assembly intermediates. Using this framework, we have implemented a Bitwise Macromolecular Assembly Simulator (BMAS). This software leverages our binary representation of intermediates to perform most crucial computational steps using bitwise operators. This allows us to perform highly efficient Gillespie-style stochastic simulations of macromolecular assembly, resulting in a general simulation approach that is orders of magnitude faster than existing methods. Our approach is efficient enough to study of a wide variety of macromolecular machines, in addition to providing a tool that should assist in the design of novel self-assembling nanomaterials.

The source code for the BMAS is available at http://github.com/deedslab

Acknowledgements

We are grateful to Michael Rowland and Ryan Suderman for many fruitful discussions.

Contents

Introduction

A large number of cellular functions are carried out by large protein complexes often referred to as “macromolecular machines" (Alberts (1998)). Famous examples include the ribosome, which is a $>$ 2.5 MDa complex consisting of over 70 proteins and 4 large RNA molecules, or the proteasome, which is a $>$ 1.5 MDa complex with over 60 component protein subunits (Korostelev and Noller (2007); Bashan and Yonath (2008); Lowe *et al*. (1995); Groll *et al*. (1997)). Most of the machines that have been characterized experimentally are not functional unless the component subunits are assembled into a specific 3-dimensional structure. Although the assembly of a small number of machines (like the ribosome and the proteasome) have been studied experimentally, there has been very little theoretical or computational investigation of macromolecular assembly pathways (Deeds *et al*. (2012a); Saiz and Vilar (2006); Murata *et al*. (2009); Marques *et al*. (2009); Williamson (2005)).

The only exception to the above statement is the assembly of viral capsids, which has been studied in some detail using both mathematical models and computational simulations (Nguyen *et al*. (2007); Rapaport *et al*. (1999); Schwartz *et al*. (1998); Jamalyaria *et al*. (2005); Zhang *et al*. (2005); Sweeny *et al*. (2008)). A number of these studies employ coarse-grained biophysical simulation techniques like Brownian Dynamics (BD) or Discrete Molecular Dynamics (DMD). While much more computationally efficient than traditional all-atom Molecular Dynamics approaches, BD and DMD are still quite computationally intensive. Simulating even a single *in vitro* assembly reaction for the proteasome, for instance, would involve tracking thousands of protein subunits over the course of hours (Zühl *et al*. (1997)), well beyond the scale of BD and DMD (Nguyen *et al*. (2007); Rapaport *et al*. (1999); Schwartz *et al*. (1998)). While these explicit techniques can consider the assembly of a single viral capsid, they are too expensive to employ as general-purpose assembly simulation tools (Nguyen *et al*. (2007); Rapaport *et al*. (1999); Schwartz *et al*. (1998)).

Given the computational expense of explicit biophysical approaches, a number of simulation methodologies have been developed that model assembly at the level of the binding interactions among possible intermediate species (Jamalyaria *et al*. (2005); Zhang *et al*. (2005); Bray and Lay

(1997); Saiz and Vilar (2006); Deeds *et al*. (2012a)). The most generalized of these “Chemical Reaction Network" (CRN) approaches is the Discrete Event Simulation of Self Assembly (DESSA) developed by Schwartz and co-workers (Jamalyaria *et al*. (2005); Zhang *et al*. (2005); Sweeny *et al*. (2008)). DESSA uses a geometrical approach to represent subunits, and is efficient enough to simulate the assembly of 10s of large viral capsids (Sweeny *et al*. (2008)). While this representation is particularly suited to simulating viral capsid geometries, extending the approach to less symmetric architectures (e.g. the ribosome) would be quite difficult.

In this work, we develop a novel CRN-based simulator for macromolecular assembly. Inspired by rule-based approaches to simulating biochemical signaling networks, this approach specifies macromolecules on the basis of molecular graphs rather than geometric constraints (Deeds *et al*. (2012b); Danos and Laneve (2004); Feret *et al*. (2009); Chylek *et al*. (2014)). As a result, our method is more directly applicable to cases with higher subunit complexity (i.e. a large number of unique subunits) or greater asymmetry than is typical of viral capsid structures (Korostelev and Noller (2007); Bashan and Yonath (2008); Lowe *et al*. (1995); Groll *et al*. (1997)). Our approach also extends the binary “presence/absence" representation pioneered by Saiz and Villar to represent possible intermediates in the assembly process (Saiz and Vilar (2006); Suderman and Deeds (2013)). This allows us to perform many key computational operation at the level of bitwise operators on binary strings. The Bitwise Macromolecular Assembly Simulator (BMAS) we have developed is thus highly computationally efficient, with gains of over an order of magnitude compared to DESSA. Our approach is also highly extensible, and is applicable to essentially any macromolecular structure of instance. In the future, the graphical basis of our representation should allow for the application of general rule-based constraints (e.g. using the Kappa language (Deeds *et al*. (2012b); Danos and Laneve (2004); Feret *et al*. (2009))) to represent the action of chaperones, transport between compartments, or other events that are common in *in vivo* assembly pathways but difficult to represent in other frameworks (Murata *et al*. (2009); Marques *et al*. (2009); Williamson (2005)).

Methods

Underlying model of assembly

As mentioned above, BMAS belongs to a class of assembly simulators that represent the assembly process at the level of chemical reactions among intermediates. In this case, we abstract a given macromolecular structure (e.g. the PDB structure on the left in Fig. 1a) as a graph where the protein subunits are the nodes and the non-covalent interactions between those subunits are represented as edges (e.g. the graph on the right in Fig. 1a). The macromolecular machines whose assembly we wish to model are generally structurally well-defined. For instance, the proteasome Core Particle (CP) is comprised of 28 subunits in four stacked seven-member rings, and structures with more subunits (say eight-member rings) have never been observed during assembly either *in vitro* or *in vivo* (Zühl *et al*. (1997); Murata *et al*. (2009); Marques *et al*. (2009)). Achieving this degree of structural specificity requires that the bond angles of the structure be sufficiently rigid; as a consequence, every possible intermediate that can be formed is a subgraph of the fully assembled structure (see Fig. 1 and Deeds *et al*. (2012a); Saiz and Vilar (2006); Bray and Lay (1997)).

Given this constraint, there are essentially 3 possible types of reactions for any given structure (Deeds *et al*. (2012a)). The first class involves the reaction of two intermediates that form a single non-covalent interaction (which we will refer to as a “bond" for simplicity), Fig. 1b. Each bond in a given structure has associated with it a given interaction affinity or K_D, specified by the user, which allows us to define the rate of the reverse reaction given a rate of association (see below). The second class of reaction occurs between two intermediates and simultaneously forms more than one bond, Fig. 1c. As detailed in the extensive supplementary material for ref. (Deeds *et al*. (2012a)), the reverse rate of these reactions is generally incredibly slow, with half-lives on the order of 10^9 s or more. While it is theoretically possible to calculate this reverse rate of this class of reactions given the K_D's of the bonds involved, these rates are so low that dissociation of ring-containing structures generally will not occur on biologically relevant timescales. As such, we consider reactions of this class to be essentially irreversible (Fig. 1c), which is a common practice

among similar modeling approaches (Deeds *et al*. (2012a); Saiz and Vilar (2006); Bray and Lay (1997); Sweeny *et al*. (2008)).

The final class of reaction that might occur is one in which the final structure that is formed would have *more* subunits than the full structure (Fig. 1d). As mentioned above, our approach is restricted to modeling the assembly of structures that are fairly rigid, and so these types of reactions (which generate a steric “clash"), are not allowed in our model.

In order to parameterize the assembly kinetics for any given structure, we must define an association rate and dissociation rate for every possible reaction. As with previous reaction-based models of macromolecular assembly (Deeds *et al*. (2012a); Saiz and Vilar (2006); Bray and Lay (1997); Sweeny *et al*. (2008)), we make the simplifying assumption that there is a uniform association rate k_f for all “allowable" reactions in the system. In other words, any dependence of the forward reaction rate on the size or identity of the particular intermediates that are binding to one another is neglected. Given this association rate, the dissociation rate for any single bond in the structure, say, between subunits “i" and “j" can be calculated as: $k_r(i,j) = k_f \times K_D(i,j)$.

Bitwise representation

In order to develop our simulation algorithm, it is helpful to introduce a formal notation regarding our representation of the macromolecule itself. As mentioned above, we abstract fully assembled structures as graphs, specifically undirected graphs $G_f = \{V_f, E_f, T_f, \tau_f\}$ where V_f is the set of vertices or nodes and E_f the set of edges between them. Many macromolecular machines are “homomeric," in that they consist of only a single type of protein (e.g. the structure in Fig. 1a), but others, like the ribosome and proteasome, may contain a set of different protein subunits that must assemble to form the given structure. To represent this, we define a set T_f of “protein types" or names for each structure, and a function $\tau_f : V_f \rightarrow T_f$ that maps the elements of V_f to their corresponding type.

Note that, according to the rules discussed in section , any intermediate in the assembly process can also be represented as a graph. Formally speaking, a “specific intermediate" in the assembly

process is an induced subgraph G_i corresponding to $V_i \subset V_f$ such that G_i comprises a single connected component. Two such specific intermediates are shown on the left in Fig. 2; note that, while these formally represent two different graphs, since the underlying structure is homomeric, these two specific intermediates represent the same physical object (i.e. a dimer). To accurately represent intermediates that arise from symmetric molecules, we say that two specific intermediates G_i and G_j are isomorphic if and only if there exists a bijective function $Q : G_i \to G_j$ such that $\tau_f(v) = \tau_f(Q(v)), \forall v \in V_i$ and $(v,w) \in V_i \iff (Q(v),Q(w)) \in V_j, \forall (v,w) \in V_i$. In other words, two specific intermediates are equivalent if we can define a bijective mapping between them that preserves protein types and edges; if so, we say $G_i \sim G_j$. We call the equivalence class of any specific intermediate a "general intermediate" (or just "intermediate" for simplicity), and we define S as the set of these general intermediates (i.e. the set of unique chemical "species"). Note that, while we refer to this as the set of intermediates, the equivalence class of the fully assembled structure G_f also belongs to this set.

To construct our bitwise representation, we first define $n \equiv |V_f|$ to be the number of total subunits in the structure. We then construct a bijective function that assigns a unique integer label to each node in V_f (see the labels 0 to 4 for the pentamer in Fig. 1a). Since there are many such bijections, we choose one arbitrarily and call ϕ the "base mapping" for G_f.

We deal with symmetry in this framework by defining a set of functions that we term "proper automorphisms". A proper automorphism is defined as a mapping $\psi : \mathbb{Z}_n \to \mathbb{Z}_n$ such that $\tau_f(\phi(\psi(i))) = \tau_f(\phi(i)), \forall i \in \mathbb{Z}_n$ and $(\phi(\psi(i)),\phi(\psi(j))) \in V_f \iff (\phi(i),\phi(j)) \in V_f, \forall (i,j) \in \mathbb{Z}_n^2$. We can think of ψ as defining a bijective function that generates an isomorphic graph, hence the term automorphism. We define the set of all proper automorphisms for G_f to be Ψ_f.

Give the above preliminary definitions, we can construct our bitwise representation of intermediates. We start by defining another type of map, $m : \mathbb{Z}_n \to \{0,1\}$. We say that a map m_i is a *representation* of a specific intermediate G_i if and only if $m_i(k) = 1 \iff \phi(k) \in V_i$ and $m_i(k) = 0 \iff \phi(k) \notin V_i, \forall k \in \mathbb{Z}_n$. In other words, m_i is a representation of G_i if it completely defines the presence/absence of nodes in G_i under the base mapping ϕ. Note that, if $G_i \sim G_j$,

then $\exists \psi \in \Psi_f$ such that $m_i(k) = 1 \iff m_j(\psi(k)) = 1$ and $m_i(k) = 0 \iff m_j(\psi(k)) = 0$, $\forall k \in \mathbb{Z}_n$. There is thus a natural equivalence class for these functions m, where every member of that class is a representation of one of a set of isomorphic specific intermediates. Given some G_i, we call the corresponding equivalence class of these functions M_i.

Although we have constructed these representations m as functions, they can equivalently be thought of as "binary vectors" of 0's and 1's, indexed by $\mathbb{Z}_n$ (see the right side of Fig. 2). It is also natural to think of m as the binary representation of some integer $x \in \mathbb{Z}$ using the usual definition $x = \sum_{k=0}^{n-1}(m(k) \cdot 2^k)$. Since they are all equivalent, we will use the above interpretations of m interchangeably. Because m can be considered an integer, for any general intermediate $s_i \in S$ we can define its canonical representation via a map $c : S \to \mathbb{Z}$ where $c(s_i) = \sup(m_i \in M_i)$. The canonical representation is thus just the largest integer that can be used to represent a given general intermediate.

It is also helpful to develop another function $Im : \mathbb{Z}_n \to \{0,1\}$ for detecting successful association reactions. We define a function Im_i as the "image" of some specific intermediate G_i in the following way: $Im_i(k) = 1$ if and only if $\phi(k) \notin V_i$ and $\exists v \in V_i$ such that $(v, \phi(k)) \in E_f$. In other words, the image of a node labeled by $k \in \mathbb{Z}_n$ under ϕ is 1 only if that node is *not present* in that specific intermediate, but at least one neighbor of that node in the graph *is present*. The image of a graph G_i thus represents the set of "unoccupied neighbors" of nodes in G_i. As with m_i, Im_i can be represented as a function, a binary vector or a binary integer. Finally, we use the following notation to denote a number of operations on our bitwise representations: & is the traditional bitwise "AND" operation; | is the traditional bitwise "OR" operation; and $\veebar$ is the traditional bitwise "XOR" operation.

Association reactions

The rules described in section can be used to define valid association reactions in this framework. An association reaction is any ordered triple of general intermediates $(r1, r2, p)$ where $\exists G_{r1} \in r1, G_{r2} \in r2$ and $G_p \in p$ such that:

1 $V_{r1} \bigcap V_{r2} = \emptyset$. This implements the “no clash" rule (Fig. 1d), since any given vertex can only exist in either of the reactants G_{r1} or G_{r2}.

2 G_p is the induced subgraph of $V_{r1} \bigcup V_{r2}$.

Note that the fact that $G_p \in p$ is a specific intermediate requires that it comprise a single connected component. As a result, $|E_p| > |E_{r1}| + |E_{r2}|$; in other words, the association reaction must form one or more bonds (Fig. 1b and c), as expected. We call the triple $(G_{r1}, G_{r2}, G_p$ a “reaction instance" for the general association reaction defined by $(r1, r2, p)$. The set of all possible reactions for any G_f is called R.

To implement this definition of an association reaction in our framework, we make the following propositions:

Proposition 1 (Association test): Say m_{r1} is the representation of a specific intermediate G_{r1}, and m_{r2} is the representation of a specific intermediate G_{r2}. Then G_{r1} and G_{r2} can associate if and only if $m_{r1} \& m_{r2} = 0$ and $m_{r1} \& Im_{r2} > 0$. If these conditions are met, then the product of the association reaction G_p has $m_p = m_{r1} \,|\, m_{r2}$.

Proof: Recall that $m_i(j) = 1$ iff $\phi(j) \in V_i$. If $m_{r1} \& m_{r2} = 0$, there is no $j \in \mathbb{Z}_n$ for which $m_{r1}(j) = 1$ and $m_{r2}(j) = 1$, so $m_{r1} \& m_{r2} = 0 \iff V_{r1} \bigcap V_{r2} = \emptyset$. Since G_{r1} and G_{r2} are specific intermediates by supposition, they both comprise a single connected component. If $m_{r1} \& Im_{r2} > 0$, that means that there is some $j \in \mathbb{Z}_n$ such that $m_{r1}(j) = 1$ and $Im_{r2}(j) = 1$. This directly implies that there is at least one pair of vertices, call them v_x and v_y, where: $v_x \in V_{r1}$, $v_y \in V_{r2}$ and $(v_x, v_y) \in E_f$. Note that, by the first condition, these vertices are unique to their corresponding subgraphs, and thus the edge between them is not present in either specific intermediate. Since G_p is defined as the *induced subgraph* of $V_{r1} \bigcup V_{r2}$, that edge is present in G_p. Thus G_p comprises a single connected component, since the edge (v_x, v_y) can exist on the path between any node from V_{r1} and any node from V_{r2} in G_p. Finally, the fact that $m_p = m_{r1} | m_{r2}$ follows directly from the definition of m and the bitwise OR operation. □

Proposition 2 (Image generation): Say that m_{r1} and m_{r2} result in a productive association reaction to form m_p. Then Im_p can be obtained from the following equation: $Im_p = ((Im_{r1} \,|\, Im_{r2}) \,|\, m_p) \veebar m_p$.

Proof: The equation can be decomposed into three operations:

Op1: Say $O_1 = Im_{r1} \,|\, Im_{r2}$. Thus, $O_1(j) = 1$ if $\phi(j)$ was an unoccupied neighbor of any node in either G_{r1} or G_{r2}.

Op2: Say $O_2 = (Im_{r1} \,|\, Im_{r2}) \,|\, m_p = O_1 \,|\, m_p$; in this case, $O_2(k) = 1$ if and only if $\phi(j)$ is an unoccupied neighbor in G_{r1} or G_{r2}, or if that node is present in G_p.

Op3: Say $O_3 = O_2 \veebar m_p$. The claim is $Im_p = O_3$. To prove this claim, consider the truth table for this operation. If $O_2(j) = 0$ and $m_p(j) = 0$, this indicates that position j is not an unoccupied neighbor in either source graph, nor is $\phi(j)$ present in G_p. So $Im_p(j) = 0$, and note that $O_3(j) = 0$ for this case. Note that, if $m_p(j) = 1 \rightarrow O_2(j) = 1$ according to Op2, which means that we can never have the case$O_2(j) = 0, m_p(j) = 1$. So, if $m_p(j) = 1$, then $O_3(j) = 0$. If a node $\phi(j) \in G_p$, then $Im_p(j) = 0$ by the definition of the image, so both $O_3(j) = 0$ and $Im_p(j) = 0$ for this case. Finally, consider the case $m_p(j) = 0$ and $O_2(j) = 1$, which will only occur when the node $\phi(j) \notin V_p$. This directly implies that $\exists v \in V_{r1} \bigcup V_{r2}$ such that $(v, \phi(j)) \in E_f$. So, $\phi(j)$ is a node that *is not* in V_p, but there is at least one node in V_p that has an edge with $\phi(j)$ in G_f. Note that this is the only condition where $O_3(j) = 1$, and also the only condition where $Im_p(j) = 1$. Since O_3 and Im_p are equal in all possible cases, $Im_p = O_3$. $\square$

Fig. 3 demonstrates the image generation operations for an example molecule. Given the proofs above, given two bitsets m_{r1} and m_{r2}, we can determine if the corresponding specific intermediates G_{r1} and G_{r2} can successfully associate with one another and, if so, we can generate the bitset for the product intermediate m_p and its image Im_p. All of the operations that are required to perform these tests and generate the product can be implemented as bitwise operators, allowing association

reactions to be treated in a computationally efficient manner.

Results

Verifying BMAS

To verify that BMAS produces results consistent with the underlying model, we compared the results of stochastic BMAS simulations with numerical integration of ODEs. We considered three distinct structures: the simple homomeric trimer, a “stacked trimer" consisting of three homomeric rings bound to one another in an a_3a_3 configuration, and a heteromeric structure with four stacked three-member rings ($a_3b_3b_3a_3$). In each case, the ODEs were generated using a completely separate enumeration algorithm as described in the beginning of section **??**; numerical integration was accomplished using the CVODE library in SUNDIALS (Hindmarsh *et al*. (2005)). We conducted a set of 10 independent stochastic simulations of each architecture using BMAS. Both types of simulations were run with the same sets of parameters. Initial conditions in both cases were set to 100% monomers, and the BMAS simulations each included enough monomers to construct 1000 fully-assembled structures.

A set of example time courses for each of the three structures can be found in Fig. 5. As one can see, agreement between these independent simulations is excellent. Note that, while these examples have been chosen with a particular set of affinities for the interactions and a particular total monomer concentration, results for a wide variety of parameters were identical to those in Fig. 5 (data not shown).

We also compared the results from BMAS to those from DESSA, using a version of DESSA obtained directly from the Schwartz lab (Sweeny *et al*. (2008)). We considered the assembly of a stacked four-member ring, similar to the structure Fig. 5b but with four subunits in each ring. As in the comparison of BMAS with the ODEs, we obtained the same results from both simulation approaches (data not shown).

Benchmarks

To benchmark BMAS, we compared its running time to that of DESSA for the stacked four-member ring structure, varying the total number of molecules in the mixture (i.e. $|E_0|$). As one can see from Fig. 6a, the running time for BMAS is over an order of magnitude smaller than DESSA across a range of system sizes. Perhaps more importantly, the scaling behavior of the two simulation approaches is quite different. BMAS exhibits a linear increase in runtime as the number of molecules in the system increases, while the scaling for DESSA is approximately quadratic on a log-log scale.

We also considered how our run time scales with the number of subunits in the structure. Focusing on the stacked ring architecture, we varied the size of G_f from the stacked 3-member ring to a stacked 40-member ring (with n varying from 6 to 80). Although the scaling of BMAS is quadratic in the number of nodes in the graph (Fig. 6b), the simulation of the stacked 40-member ring, which involves simulating the assembly of 1000 molecules for over 3 hours of simulated time, takes less than 15 minutes on a laptop computer. The results shown in Fig. 6 thus highlight the computational efficiency of our bitwise approach.

Discussion

As discussed in the introduction, the problem of macromolecular assembly is foundational to our understanding of a host of cellular processes. While experimental studies have provided some insights into assembly pathways for a small number of example molecules (e.g. the ribosome or proteasome), the inherently coarse-grained nature of experimental observables (particularly *in vivo*) requires the development of computational simulations to complement experimental approaches (Murata *et al*. (2009); Marques *et al*. (2009); Williamson (2005); Deeds *et al*. (2012a); Sweeny *et al*. (2008)). The timescales of assembly processes, however, are sufficiently long that traditional biophysical approaches like BD and DMD are not readily applicable (Schwartz *et al*. (1998); Nguyen *et al*. (2007); Rapaport *et al*. (1999)). As a result, there is a clear need for the development

of efficient, scalable simulation approaches for macromolecular assembly problems.

Although DESSA represents one such approach, its geometric representation is somewhat cumbersome, and it is not easily extensible to the wide variety of structural architectures observed for macromolecular machines inside cells (Levy *et al*. (2006)). In this work, we have developed an alternative approach that is based on a graphical, rather than geometric, representation of macromolecular structures (Fig. 1). This representation allows us to develop a formally rigorous and computationally efficient bitwise representation of intermediates. The Bitwise Macromolecular Assembly Simulator that we have implemented achieves gains in computational efficiency of over an order of magnitude compared with existing approaches (Fig. 6), allowing it to be applied to a wide variety of very large complexes. It is also highly extensible, in that the input to the simulation is a representation of the molecular graph of the complex, which is readily obtained from PDB structures (Levy *et al*. (2006)).

Given its graphical representation of intermediates, BMAS is naturally compatible with existing rule-based approaches to specifying computational models of signaling networks (Deeds *et al*. (2012b); Danos and Laneve (2004); Feret *et al*. (2009)). We envision that combining BMAS with a rule-based language will allow for the specification of rules that represent mechanisms like allosteric communication among subunits, transport among intracellular compartments, and chaperone mechanisms, all of which would be difficult (if not impossible) to implement in more traditional modeling frameworks. This will allow for the application of BMAS to the study of generalized assembly mechanisms within cells, as well as for the design of efficient assembly pathways for novel synthetic self-assembling systems.

Bibliography

Alberts, B. (1998) The cell as a collection of protein machines: preparing the next generation of molecular biologists, *Cell*, **92** (3), 291-2944.

Bashan, A. and Yonath, A. (2008) Correlating ribosome function with high-resolution structures, *Trends Microbiol*, **16** (7), 326-335.

Bray, D. and Lay, S. (1997) Computer-based analysis of the binding steps in protein complex formation, *Proc Natl Acad Sci USA*, **94** (25), 13493-13498.

Chylek, L. A. *et al*. (2014) Rule-based modeling: a computational approach for studying biomolecular site dynamics in cell signaling systems, *Wiley Interdiscip Rev Syst Biol Med*, **6** (1), 13-36.

Danos, V. and Laneve, C. (2004) Formal molecular Biology, *Theoretical Comp Sci*, **325** (1), 69-110.

Deeds, E. J., Bachman, J. A. and Fontana, W. (2012a) Optimizing ring assembly reveals the strength of weak interactions, *Proc Natl Acad Sci USA*, **109** (7), 2348-2353.

Deeds, E. J. *et al*. (2012) Combinatorial complexity and compositional drift in protein interaction networks, *PLoS ONE*, **7** (3), e32032.

Feret, J. *et al*. (2009) Internal coarse-graining of molecular systems, *Proc Natl Acad Sci USA*, **106** (16), 6453-6458.

Groll, M. *et al*. Structure of 20S proteasome from yeast at 2.4 Å resolution, *Nature*, **386** (6624), 463-471.

Hindmarsh, A. C. *et al*. (2005) SUNDIALS: suite of nonlinear and differential/algebraic equation solvers, *ACM Trans Math Software*, **31** (3), 363-396.

Jamalyaria, F., Rohlfs, R. and Schwartz, R. (2005) Queue-based method for efficient simulation of biological self-assembly systems, J Comput Phys, **204** (1), 100-120.

Korostolev, A. and Noller, H. F. (2007) The ribosome in focus: new structures bring new insights, *Trends Biochem Sci*, **32** (9), 434-441.

Levy, E. D. *et al*. (2006) 3D complex: a structural classification of protein complexes, *PLoS Comput Biol*, **2** (11), e155.

Lowe *et al*. (1995) Crystal structure of the 20S proteasome from the archaeon *T. acidophilum* at 3.4 Å resolution, *Science*, **268** (5210), 533-539.

Marques, A. J. *et al*. (2009) Catalytic mechanism and assembly of the proteasome, *Chem Ref*, **109** (4), 1509-1536.

Murata, S., Yashiroda, H. and Tanaka, K. (2009) Molecular mechanisms of proteasome assembly, *Nat Rev Mol Cell Biol*, **10** (2), 104-115.

Nguyen, H. D., Reddy, V. S. and Brooks, C. L. (2007) Deciphering the kinetic mechanism of spontaneous self-assembly of icosahedral capsids, *Nano Lett*, **7** (2), 338-344.

Rapaport, D.C., Johnson, J. E. and Skolnick, J. (1999) Supramolecular self-assembly: molecular dynamics modeling of polyhedral shell formation, *Comput Phys Commun*, **121**, 231-235.

Saiz, L. and Vilar, J. M. G. (2006) Stochastic dynamics of macromolecular-assembly networks, *Mol Syst Biol*, **2**, 2006.0024.

Schwartz, R. *et al*. (1998) Local rules simulation of the kinetics of virus capsid self-assembly, *Biophys J*, **75** (6), 2626-2636.

Suderman, R. and Deeds, E. J. (2013) Machines vs. ensembles: effective MAPK signaling through heterogeneous sets of protein complexes, *PLoS Comput Biol*, **9** (10), e1003278.

Sweeny, B., Zhang, T. and Schwartz, R. (2008) Exploring the parameter space of complex self-assembly through virus capsid models, *Biophys J*, **947** (3), 772-783.

Williamson, J. R. (2005) Assembly of the 30S ribosomal subunit, *Q Rev Biophys*, **38** (04), 397-403.

Zhang, T., Rohlfs, R. and Schwartz, R. (2005) Implementation of a discrete event simulator for biological self-assembly systems, *Proceedings of the 2005 Winter Simulation Conference*.

Zühl *et al*. (1997) Dissecting the assembly pathway of the 20S proteasome, *FEBS Lett*, **418** (1-2), 189-194.

Figure captions

Figure 1: The underlying assembly model. (a) We abstract the 3D structure of homomeric pentamer (on the left) as a graph, where each node is one subunit and each edge represents the non-covalent interaction or bond between two adjacent subunits. (b) An example of a “class 1" reaction, where a single bond is formed by the association of a monomer and a dimer to form a trimer. (c) An example of a “class 2" reaction, where a dimer and a trimer react to form the full pentamer. Note that two bonds are formed simultaneously in this case, and we consider such reactions to be essentially irreversible in our model of assembly. (d) In this case, the two intermediates cannot react since doing so would either generate a structure with a total of six subunits.

Figure 2: Bitwise Representation of Macromolecules. The graphs on the left in (a) and (b) are a representation of two specific intermediates; nodes that are present in the intermediate are colored in, whereas nodes that are absent are white. Note that, since the pentamer in question is homomeric, the two graphs are isomorphic, and thus belong to the same general intermediate. The bitwise representation for each specific intermediate is denoted on the right side of the (a) and (b),

with the dashed arrows representing the “base mapping" ϕ. Since the graphs in (a) and (b) are isomorphic, there is a proper automorphism between m_a and m_b (purple arrow).

Figure 3: Bitwise operations in association reactions. (a) This represents the association test. On the left is bitwise “AND" on two molecules A and B; note that, since this operation evaluates to “0," there is no clash in this case. On the right is the bitwise “AND" of molecule A and image of B. Since this operation evaluates to a binary number that is greater than 0, the reaction between A and B will form edges. (b) This is an example of the set of operations that generate the image associated with the new molecule C obtained by the reaction of A with B. Op1, denoted by the first “OR" operation, results in a bitset that is “1" for all possible positions that could be unoccupied neighbors in C. Op2, denoted by the second OR operation, generates a bitset that is 1 at every position that is either an unoccupied neighbors in C, or is actually present in C. The final operation, Op3, uses a bitwise “XOR" to generate the image of C.

Figure 4: Flow chart for our stochastic simulation of macromolecular assembly

Figure 5: Validation of BMAS. As described in the text, we enumerated the ODEs for the Trimer, Stacked Trimer and Four Layer Trimer structures. The three curves represent example simulations comparing the results from stochastic simulation with BMAS to the results of numerical integration by ODEs. Each curve represents the fraction of fully-assembled structures formed in each system as a function of time, with time plotted on a logarithmic scale. The error bars represent the standard deviation among 10 independent BMAS runs.

Figure 6: Running time benchmarks. (a) This is a comparison of running time as a function of system size (defined as the number of possible fully assembled molecules) for DESSA and BMAS. Simulations in this case were run on a stacked tetramer, a structure similar to that in Fig. 5b but with four rather than three subunits in each ring. Note that BMAS is generally an order of magnitude more efficient, and that its scaling is linear, whereas the scaling of DESSA is approximately quadratic on a log-log scale. (b) This represents the running time of BMAS as a function of size of the ring in a stacked ring macromolecule.

Figures

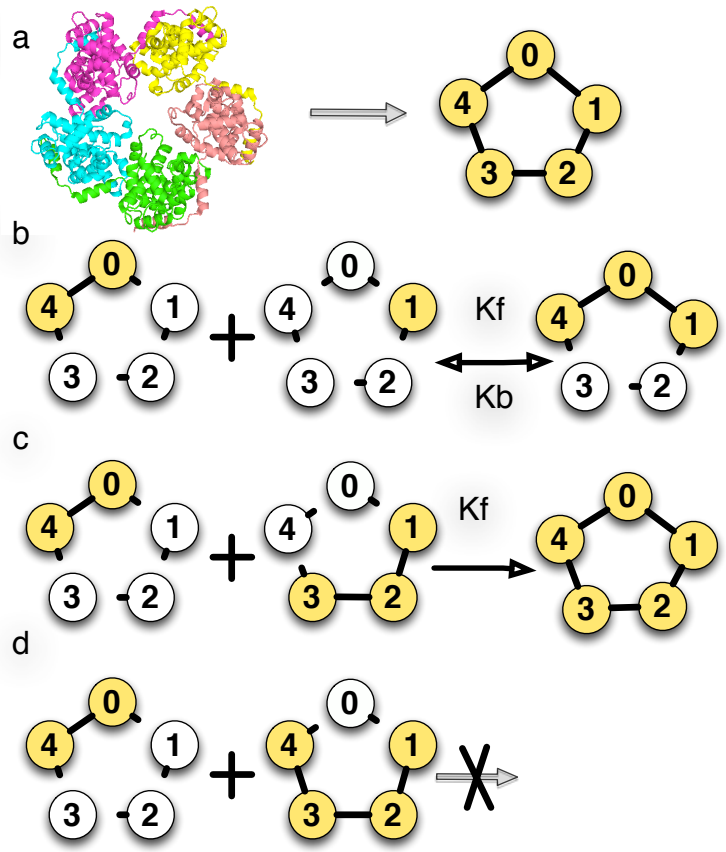

Figure 1

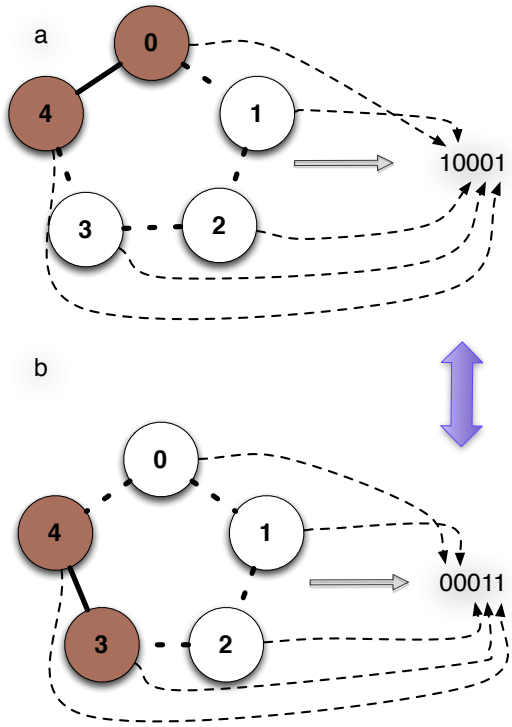

Figure 2

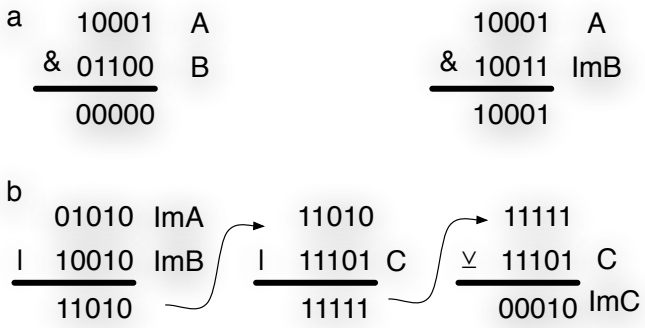

Figure 3

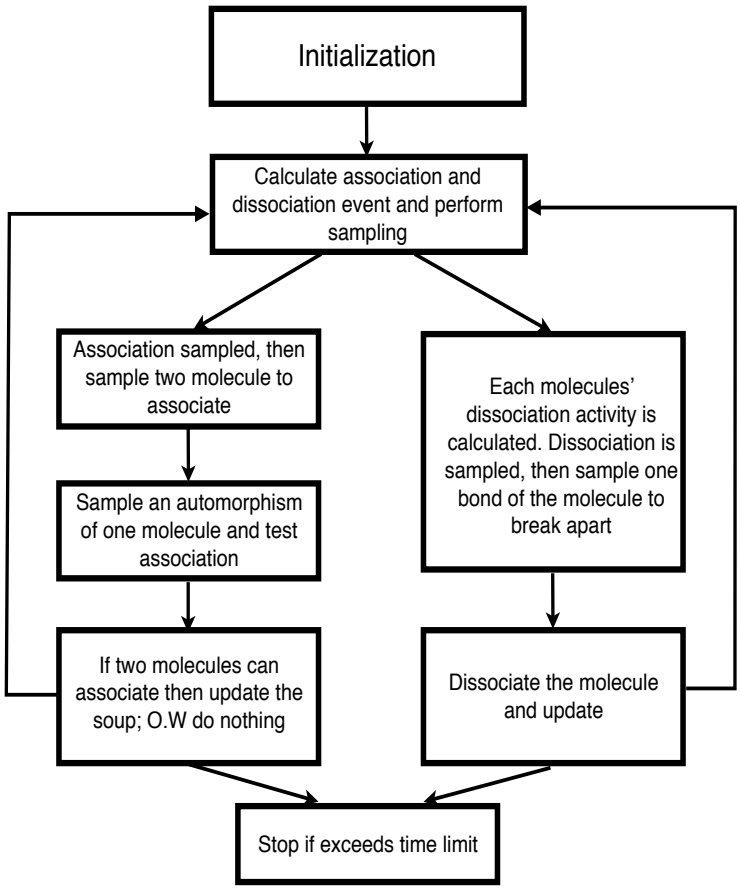
Initialization
Calculate association and dissociation event and perform sampling
Association sampled, then sample two molecule to associate
Sample an automorphism of one molecule and test association
If two molecules can associate then update the soup; O.W do nothing
Each molecules' dissociation activity is calculated. Dissociation is sampled, then sample one bond of the molecule to break apart
Dissociate the molecule and update
Stop if exceeds time limit

Figure 4

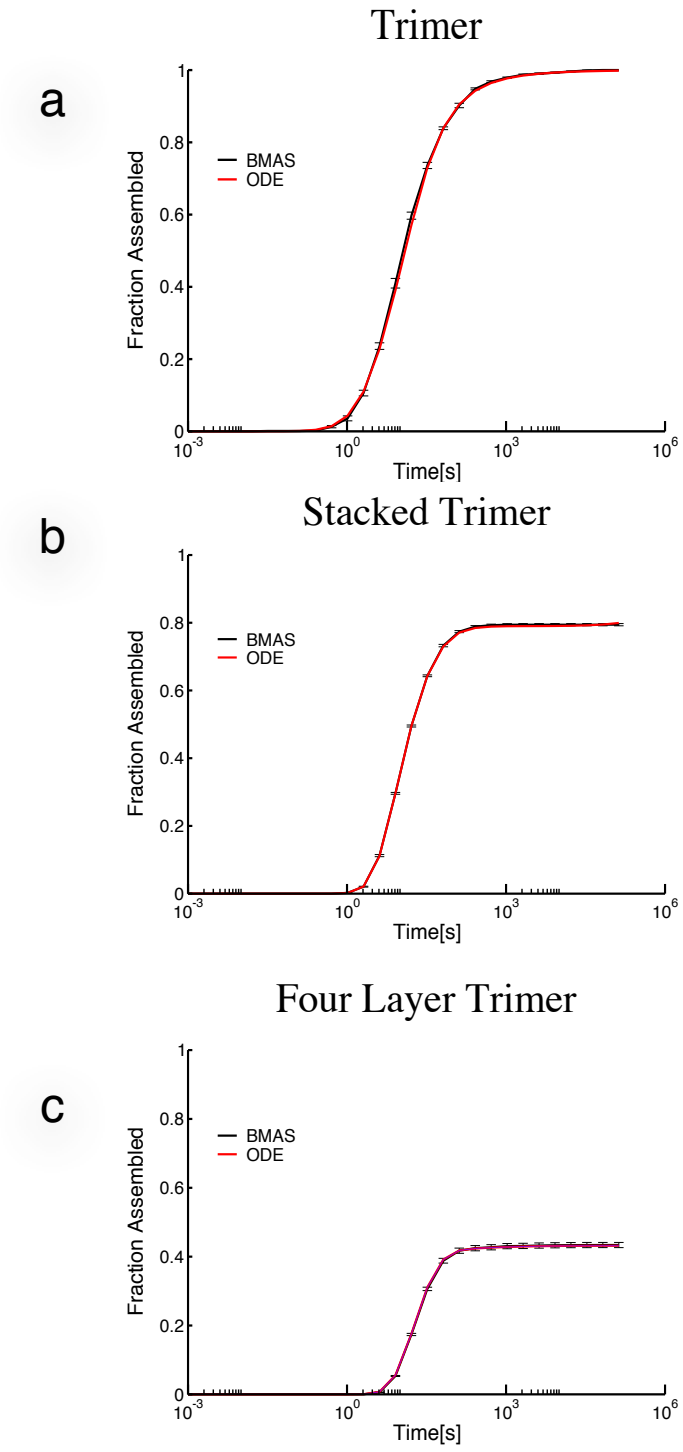
a
Trimer
Fraction Assembled
BMAS
ODE
Time[s]
b
Stacked Trimer
Fraction Assembled
BMAS
ODE
Time[s]
c
Four Layer Trimer
Fraction Assembled
BMAS
ODE
Time[s]

Figure 5

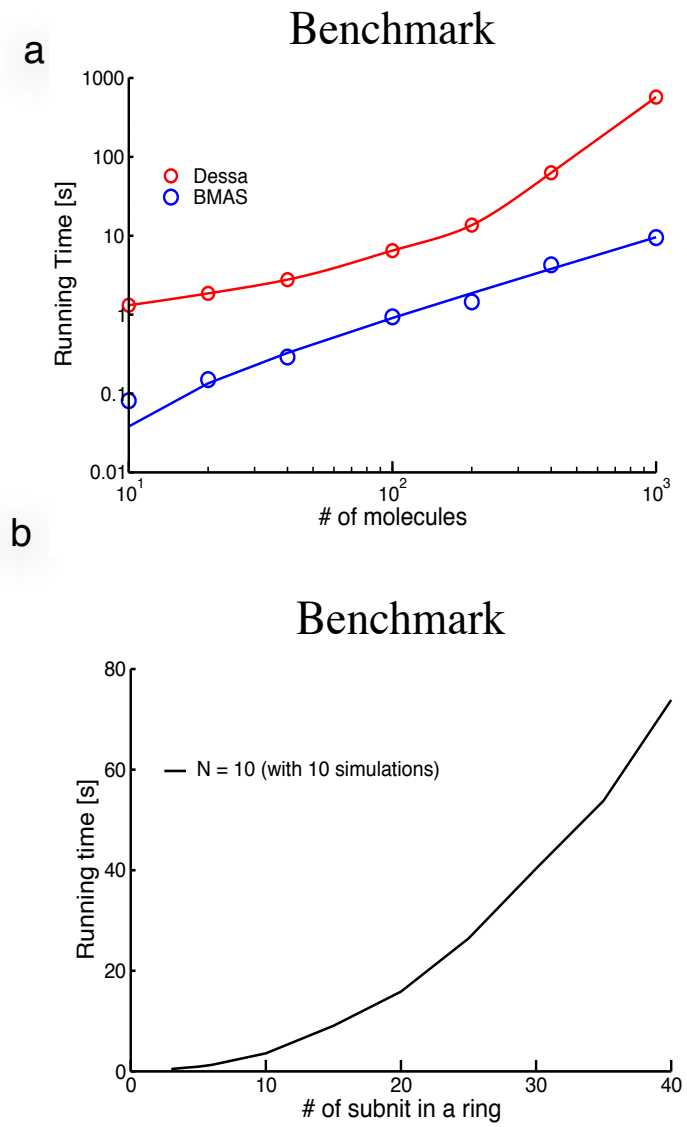
a
Benchmark
1000
100
10
1
0.1
0.01
Running Time [s]
Dessa
BMAS
10^1
10^2
10^3
# of molecules
b
Benchmark
80
60
40
20
0
Running time [s]
N = 10 (with 10 simulations)
0
10
20
30
40
# of subnit in a ring

Figure 6