Novel domain expansion methods to improve the computational efficiency of the Chemical Master Equation solution for large biological networks

Rahul Kosarwal$^{1,2}$, Don Kulasiri$^{1,2}$* and Sandhya Samarasinghe$^{1,2}$

*Correspondence: Don.Kulasiri@lincoln.ac.nz

**Methodology Article**

**Abstract**

**Background:** Numerical solutions of the chemical master equation (CME) are important for understanding the stochasticity of biochemical systems. However, solving CMEs is a formidable task. This task is complicated due to the nonlinear nature of the reactions and the size of the networks which result in different realizations. Most importantly, the exponential growth of the size of the state-space, with respect to the number of different species in the system makes this a challenging assignment. When the biochemical system has a large number of variables, the CME solution becomes intractable. We introduce the intelligent state projection (ISP) method to use in the stochastic analysis of these systems. For any biochemical reaction network, it is important to capture more than one moment: this allows one to describe the system's dynamic behaviour. ISP is based on a state-space search and the data structure standards of artificial intelligence (AI). It can be used to explore and update the states of a biochemical system. To support the expansion in ISP, we also develop a Bayesian likelihood node projection (BLNP) function to predict the likelihood of the states.

**Results:** To demonstrate the acceptability and effectiveness of our method, we apply the ISP method to several biological models discussed in prior literature. The results of our computational experiments reveal that the ISP method is effective both in terms of the speed and accuracy of the expansion, and the accuracy of the solution. This method also provides a better understanding of the state-space of the system in terms of blueprint patterns.

**Conclusions:** The ISP is the de-novo method which addresses both accuracy and performance problems for CME solutions. It systematically expands the projection space based on predefined inputs. This ensures accuracy in the approximation and an exact analytical solution for the time of interest. The ISP was more effective both in predicting the behavior of the state-space of the system and in performance management, which is a vital step towards modeling large biochemical systems.

**Keywords:** Biochemical reaction networks, chemical master equation, stochastic, intelligent state projection, Bayesian likelihood node projection

© The Author(s). 2020. Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.
Background

In systems biology, it is crucial to understand the dynamics of large and complicated biochemical reaction networks. Recent advances in computing and mathematical techniques mean it is easier for biologists to deal with enormous amounts of experimental data, right down to the level of a single molecule of a species. Such information reveals the presence of a high level of stochasticity in the networks of biochemical reactions. In biochemical reaction networks, stochastic models have made significant contributions to the fields of systems biology [1, 2], neuroscience [3], and drug modeling [4].

In a complex system, biochemical reactions are often modeled as reaction rate equations (RREs) using ordinary differential equations (ODEs). Examples of this kind of work include the biochemical networks of Alzheimer’s disease (AD) [5]; the pathways in the fungal pathogen Candida albicans [6]; and the COVID-19 coronavirus pathogen network [7]. In each of these examples, the behavior of different pathways is still largely unknown. All these models only contain species with small copy numbers and widely different reaction rates; the probabilistic descriptions of time evolution of molecular concentrations (or numbers) are more suited for understanding the dynamics of such systems. One probabilistic approach for modeling a biochemical reaction network is to deduce a set of integro-differential equations known as chemical master equations (CMEs) [8, 9]. CMEs describe the evolution of the probability distribution over the entire state-space of a biochemical system that jumps from one set of states to another set of states in continuous time: they are a continuous time version of Markov chains (CTMCs) [8, 10] with discrete states. By defining the Markov chain [10, 11], we can consider the joint and marginal probability densities of the species in a system that changes over time [12].

In such cases, the development of RREs with molecular numbers becomes very important. The biochemical reaction network can be defined in terms of the discrete state \( X \equiv (x_1, ..., x_N)^T \) vector of non-negative integers \( x_N \) for the given conditions, where \( N \geq 1 \). \( \{X(t) : t \in K; \phi\} \) defines a stochastic process, where \( K \) is the indexing scheme and \( \phi \) is the sample space. Following the derivation in [9], for every reaction, there exists a reaction channel, \( R_M \), which determines the unique reaction in the system with a propensity function \( k_M \). The specific combinations of the reactant species in \( R_M \) will react during an infinitesimal \([t, t+dt]\) time interval. The average probability \( a_\mu(X(t))dt \) of a particular \( R_M \) fires within \([t, t+dt]\) is the multiplication of the numbers of reactant species, denoted by square brackets, by \( k_M \). For example,

\[
\begin{align*}
R_1: C & \xrightarrow{k_1} A, & a_1k_1([C]) \\
R_2: A + E & \xrightarrow{k_2} T, & a_2k_2([A],[E]) \\
R_3: T & \xrightarrow{k_3} C, & a_3k_3([T])
\end{align*}
\]

In the case where the reactants are of the same type, for example \( A + A \xrightarrow{k_2} T \), then \( a_2k_2([A][A-1]/2) \). The set consisting of all the reaction channels, \( R_M \), is the union of sets of fast reactions and slow reactions [12]. They are categorized into sets of \( R_{M(f)} \) and \( R_{M(s)} \) reactions, respectively, based on their propensity values. Therefore,

\[
R_M = R_{M(f)} \cup R_{M(s)}.
\] (1)
A reaction is faster than others if its propensity is of several orders of magnitude larger than the other propensity values (see the list of abbreviations and notations at the end).

Chemical master equation

In this paper, we consider a network of biochemical reactions at a constant volume. The network consists of $\tilde{N} \geq 1$ different species $\{S_1, \ldots, S_{\tilde{N}}\}$. They are spatially homogeneous and interact through $M \geq 1$ reaction channels in thermal equilibrium. The number of counts of each different species defines the state of the system. If all the species are bounded by $S$, then the approximate number of states in the system would be $S^{\tilde{N}}$ [13].

For every state, $X$, the probability satisfies the following CME [8],

$$\frac{\partial P^{(t)}(X)}{\partial t} = \sum_{\mu=1}^{M} a_{\mu}(X - v_{\mu})P^{(t)}(X - v_{\mu}) - \sum_{\mu=1}^{M} a_{\mu}(X)P^{(t)}(X)$$ \hspace{1cm} (2)

where $P^{(t)}(X) = \text{the probability function, representing the time-evolution of the system,}$

$M = \text{elementary chemical reaction channels } R_1, \ldots, R_M,$

$a_{\mu} = \text{chemical reaction propensity of channel } \mu = \{1, 2, \ldots, M\},$ and

$v_{\mu} = \text{the stoichiometric vector that represents a change in the molecular population of the chemical species due to the occurrence of one } R_M \text{ reaction.}$

The system transitions to a new state: $X + v_{\mu}$ records the changes in the number of counts of different species when the reactions occur.

We note that $a_{\mu}(X - v_{\mu})dt$ is the probability for state $(X - v_{\mu})$ to transition to state $X$ through chemical reaction, $R_M$, during $[t, t + dt]$, and $\sum_{\mu=1}^{M} a_{\mu}(X)dt$ is the probability for the system to shift from state $X$ as a result of any reaction during $dt$. If $X_J = \{X_1,$

$\ldots, X_{\tilde{N}}\}$ is the ordered set of possible states of the system indexed by $\{1, 2, \ldots, K\}$ having $S^{\tilde{N}}$ elements, then Eq. (2) represents the set of ordinary differential equations (ODEs) that determines the changes in probability density $P^{(t)} = (P^{(t)}(X_1), \ldots, P^{(t)}(X_{\tilde{N}}))^T$.

Once $X_J$ is selected, the matrix-vector form of Eq. (2) is described by an ODE:

$$\frac{\partial P^{(t)}(X)}{\partial t} = A.P^{(t)},$$ \hspace{1cm} (3)

where the transition rate matrix is $A = [a_{i,j}]$. If each reaction leads to a different state, $X_{i,j}$, then the elements in submatrix $A_{i,j}$ are given as:

$$A_{i,j} = \begin{cases} -\sum_{\mu=1}^{M} a_{\mu}(X_i), & \text{if } i = j \\ a_{\mu}(X_i), & \text{if } X_{i,j} = X_i + v_{\mu} \\ 0, & \text{otherwise} \end{cases}$$ \hspace{1cm} (4)

This equation represents the infinitesimal generator of the Markov process [10, 14, 15]. Rows and columns are ordered in lowercase letters, $i$ and $j$ respectively. The entry of $a_{i,j}$ of the matrix determines the propensity for the chemical system to transition
from one state to another state, given that \( i \neq j \), are non-negative. The diagonal terms of the matrix are defined by \( a_{jj} \), when \( i = j \) and the matrix has a zero-column sum, so its probability is conserved. From Eq. (3) we can derive the \( P^{(t)} \) probability vector at the final time, \( t_f \) of interest given an initial density of \( P^{(0)} \):

\[
P^{(t)} = \exp(t_f A) P^{(0)},
\]

where the matrix exponential function is defined by the convergent Taylor series as

\[
\exp(t_f A) = I + \sum_{n=1}^{\infty} \frac{(t_f A)^n}{n!}.
\]

However, algorithms, such as in [13, 18–20] truncate Eq. (6) infinite summation to approximate Eq. (3) at the cost of a truncation error.

**Initial value problem**

If \( v_\mu \) or \( v_M \) for \( \mu \) or \( M = \{1, 2, ..., M \} \) are the stoichiometric vectors for \( R_M \) reaction channels, then we will define the stoichiometric matrix for the system by \( V_\mu \) or \( \Phi_M \) as \( [v_1; v_2; ...; v_\mu]^T \). If \( \phi \) is the initial state and \( X_\phi \) is the initial density of the system, \( X_j \) denotes the only set of states in \( \phi \). To solve \( P^{(t)} (X) \) in Eq. (2) for \( X \in \phi \), we define the \( P^{(t)} \) vector as \( \left( P^{(t)} (X) \right)_{X \in \phi} \), for a finite set of states, then \( \frac{\partial P^{(t)}}{\partial t} \) is defined as a vector \( \left( \frac{\partial P^{(t)}}{\partial t} \right)_{X \in \phi} \). Solving the CME involves finding the solution of the initial value problem over a time period using the differential equation Eq. (3) when \( t > 0 \), whereas, \( P^{(t)} \) is the initial distribution at \( t = 0 \). Here, the sample space \( \phi \) can be infinite for large biochemical systems. Finding the solution for Eq. (3) for the given parameters with a finite set of states \( X_j \) is a major problem for CME’s because in large biochemical systems the size of \( A \) will be extremely large.

For example, consider an enzymatic reaction network [13] described by reactions \( R_1 \):

\[
S + E \xrightarrow{k_1} C, \quad R_2: C \xrightarrow{k_2} S + E, \quad R_3: P \quad E \xrightarrow{k_3} C, \quad P \quad .
\]

This network of reactions involves four species: namely, \( S \)– substrate, \( E \)– enzyme, \( C \)– complex and \( P \)– product molecules. The \( X = (x_1, x_2, x_3, x_4)^T \equiv (S, E, C, P)^T \) represents any state of the system, with \( X_0 = (S_0, E_0, C_0, P_0) \) given as the initial state. The stoichiometric vectors are given by \( v_1 = (-1, -1, 1, 0) \), \( v_2 = (1, 1, -1, 0) \), \( v_3 = (0, 1, -1, 1) \). Therefore, for \( (x_1, x_2, x_3, x_4) \) \( x_4_N = 4 \), the propensity functions are:

\[
\begin{align*}
R_1 : a_1([x_1], [x_2], [x_3], [x_4]) &= k_1 \times x_1(t) \times x_2(t) \\
R_2 : a_1([x_1], [x_2], [x_3], [x_4]) &= k_2 \times x_2(t) \\
R_3 : a_1([x_1], [x_2], [x_3], [x_4]) &= k_3 \times x_3(t)
\end{align*}
\]

The set of states reachable from \( X_0 \) is finite in number. With multiple explosions of the number of states in a large model, the size of \( A \) increases exponentially.

As seen in Eq. (5), solving Eq. (2) becomes a problem when the model’s dimensions grow due to the increase of species present in the system. This is particularly true for large biochemical models. The approximate estimate of \( S^N \) shows how the size of the problem increases. This explosion in size is known as the curse of dimensionality [9, 13]. The CME solution given in Eq. (5) has two major parts: (a) the expansion of the
state-space, and (b) the approximation of the series. For the expansion of state-space, Finite State Projection (FSP) [21] and Sliding Windows (SW) [18] are used to find the domain. Methods like Krylov subspace [13] and Runge Kutta [22] are commonly used for approximation (of the series) of the CME Eq. (5).

Although CME has been employed and solved explicitly for relatively small biological systems [13, 18–20, 23, 24], computationally complaisant but accurate solutions are still unknown for most significant systems and for large systems which have an infinite (or a very large) number of states. This lack of closed-form solutions has driven the system biology research towards Monte-carlo Algorithms (MC) [25] to capture dynamics. One algorithm, the Stochastic Simulation Algorithm (SSA) by Gillespie [9], has been used in the CME. Although the original FSP state-space expansion has been used in research [21, 26], it has some drawbacks [21]. The FSP [21] and its variants [20, 24, 26, 27] are based on r-step reachability [26]. While SW [18] is also a FSP based method, it employs a stochastic simulation algorithm (SSA) to find the domain. This is more effective than FSP and suitable for stiff problems. Add-on weighting functions like GORDE [28] and likelihoods [24] methods are used to improve the expansion. FSP GORDE [28] removes the states with small probabilities before the calculation of Eq. (5). This practice saves computational time, meaning that FSP GORDE performs faster than conventional FSP r-step reachability. However, removing the probabilities before the calculation of Eq. (5) increases the steps error and affects the accuracy of the final solution at tf regardless of whether the state-space is small or large. If one is interested in solving stiff and/or large systems, it will greatly affect the solution.

The FSP variant, Optimal Finite State Projection (OFSP), [20] based on r-step reachability, performs better in terms of producing optimal order domain. It is faster than both FSP and FSP GORDE. However, it is infeasible to use SW for large CME problems because creating hyper-rectangles is a very difficult task. At least four-times the number of SSA simulations are required to minimize the error by half, because of very low convergence rates of routines in MC. The original SSA takes a long time, because one simulation may have several different RM. Recently, the SSA’s efficiency has been greatly enhanced by researchers through various schemes such as τ leaps (adaptive) [29, 30]. Thus, we compare the OFSP and SSA (τ leaps adaptive) against the ISP in terms of finding the domain, accuracy and computational efficiency. Key to solving the CME remains in finding the right projection size (domain) for large models which would then ensure efficient approximation.

In this paper, we focus primarily on developing the expansion strategy, namely the Intelligent State Projection (ISP) method, to mitigate several problems: the accuracy of the solution, the performance of the method and projection size. The ISP has two variants: the Latitudinal Search (LAS) and the Longitudinal-Latitudinal Search (LOLAS). It treats the Markov chain of a biochemical system as a Markov chain tree structure and states as objects of class node. Based on the dimension of the system, search is performed in a latitudinal way for different model sizes using the ISP LAS method. Whereas, bidirectional search is applied using ISP LOLAS, which quickly expands the state-space up to a specified bound limit. To support the expansion strategy, we also develop the Bayesian Likelihood Node Projection (BLNP) function, based on Bayes’ theorem [31, 32]. It is adjoined with
the **ISP** variants to determine the likelihood of events at any interval at the molecular population level. **BLNP** provides confidence to the expansion strategy by assigning probability values to the occurrence of future reactions and prioritizing the direction of expansion. The **ISP** embedding **BLNP** function inductively expands the multiple states with the likelihood of occurrence of fast and slow reactions \cite{12}. It also defines the complexity of the system by predicting the pattern of state-space updation, and the depth of the end state from the initial state. When used for any size of biological networks, **LAS’** memory usage is proportional to the entire width of expansion; it is less than **ISP LOLAS**. Both methods are feasible and differentiated for various types of biological networks. However, the computational time for both variants depend on the nature of the model and the size of the time step used. At any point, the amount of memory in use is directly proportional to the neighboring states reachable through a single **RM** reaction. **ISP LOLAS** uses considerably less memory, even when it retracts to the initial node to track new reactions, then revisiting the depth many times.

**Results**

Having discussed the CME solution, we now discuss the modeling and integration of the biochemical reaction systems for the **ISP** methods, as well as the assumptions underlying these methods. Using **ISP**, we tested its ability to reproduce the model to measure dynamics of the key parameters in the models. The **ISP** method is a novel, easy-to-use, technique for modeling and expanding the state-space of biochemical systems. It features several improvements in modeling and computational efficiency.

The computational experiment (initializing and solving the model) was conducted on the carbon-neutral platform of Amazon® Web Service Elastic Computing (EC2), instance type large (m5a), running on HVM (hardware virtual environment) virtualization with variable ECUs. We used multicore environment 16vCPU @ 2.2GHz, AMD EPYC 7571 running Ubuntu 16.04.1 with relevant dependencies, and 64GB memory with 8GB Elastic Block Storage (EBS) type General Purpose SSD (GP2) formatted with Elastic File System (EFS). The performance mode was set to General Purpose with input-output per second (IOPS = 100/3000). We used the type bursting throughput mode (see Supplementary Information (SI) 1).

**Intelligent state projection**

The main aim of the proposed algorithm is to expand the $X_K$ iteratively, such that $X_K$ contains a minimum number of states carrying the maximum probability mass of the system. To create the sample space for **ISP**, a Markov chain tree $I_K$ \cite{33} was used to visualize a biochemical system to exhibit the transition matrix as directed trees \cite{10, 11} of its associated graph. Additionally, the Markov chain tree $I_K$ generates sample space of the system to represent Markov processes associated with the Markov chain and the transition matrices of biochemical reaction networks. In following section, we visualize the Markov chain of the biochemical system as a Markov chain graph (tree) for **ISP** compatibility.
Markov chain as a Markov chain tree

We define the Markov chain tree, $\mathcal{K}$, as infinite and locally finite. It is a special type of graph with a prominent vertex called a parent node without loops or cycles. If graph $G_{mc}$ is a state-space of the finite state Markov chain with $\{P(X_i, X'_i) \mid X_i, X'_i \in G_{mc}\}$ transition probabilities meeting the condition $\sum_{X'_i} P(X_i, X'_i) = 1$, then the induced Markov chain tree is a combination of valued random variables with the distributions inductively defined from $P(X_i, X'_i)$ with an initial state, $X_i \in G_{mc}$. That being the case, it is easy to expand this class of Markov random field through a Markov chain tree structure for biochemical systems. Furthermore, the Markov chain tree and the Markov processes can be equated as explained in [34] for the stochastic analysis.

Since we are interested in aperiodic states in the expansion of state-space, we shall assume the reducibility or simplification of the $G_{mc}$; namely for each $X_i, X'_i \in G_{mc}$ through $\mathcal{K}$. Therefore, let us concentrate on the case where $G_{mc}$ is considered as a locally finite connected graph. The transition probabilities of each state are not equal due to the propensities and parameters of different reactions in the biochemical system. Consequently, a Markov chain tree, $\mathcal{K}$, can be used to visualize a biochemical system process to exhibit a transition matrix as directed trees of its associated graph [10, 11]. It can also be used to generate a sample space for the system to represent the Markov processes and the transition matrices of biochemical reaction networks. We discuss the details needed to represent Markov models on trees and working with graphs for state-space later.

If $X_f$ is the finite set of cardinality $\{1, 2, \ldots, K\}$ of a Markov chain $\mathcal{X}_c$, then $A$ is the transition probability matrix associated with $X_f$. A state-space is, substantially, a class of a set of states containing the unique state of the system. The arcs between the states represent the transitions from the initial state to the end state. This transition is defined as transient and communicating class in graphs. When all the transitions are combined, every state-space takes the form of a graph and creates the state-space of the system, as shown in Fig. 1 below.

We can now associate chain $\mathcal{X}_c$ with the directed graph $G_{mc} = (X_f, V_\mu)$, where $V_\mu = [v_1; v_2; \ldots; v_\mu]$. $v_\mu$ defines the transition from state $X_i$ to $X'_i$ and is denoted as $v_\mu = \{(X_i, X'_i) \mid a_{i,j} > 0\}$. For every transition $(X_i, X'_i)\in X_f$, then weight $\omega(X_i, X'_i)$ is $a_{i,j}$.

Suppose $G_{mc}$ has a cycle, which starts and terminates at some state, $X_i \in X_f$. If there is a transition from $X_i$ to $X'_i$, we add a unique transition by creating a cycle from $X'_i$ back to itself and then consider the original transition from $X_i$ to $X'_i$. This contradicts the uniqueness of the walk in tree [35]. In terms of the CTMC of a biochemical system process, the change in molecular population is defined by a stoichiometric vector, so, in $G_{mc}$, there must be at least one intermediate state that will send the system back to the previous state to create the cycle. This process categorizes the forward and backward reactions given the initial state, $X_0$, of the system. The transient class of the transition leads the system to a unique state that defines the forward reaction in the system. In contrast, the communicating class of a transition defines the reversible reaction in the system. We define such systems as transient class systems and communicating class systems. Large biochemical systems are usually a combination of both classes.

A biochemical system is visualized as a tree $\mathcal{K}$ to enable the expansion of the state-space. A tree, $\mathcal{K}$, is a special form of graph in data structure constituting a set of
nodes and a collection of edges (or arcs), each of which connects to an ordered pair of nodes. \(G_{mc}\) is considered a directed tree, \(\mathcal{K}\). It is rooted with \(N_0 = (X_0, \delta_0)\) if it contains a unique walk to \(N_i = (X_i, \delta_i + 1)\) and does not contain any cycles. Meanwhile, if \(X_i \in X_J \setminus \{X_0\}\) has exactly one outgoing transition away from \(X_0\) it is called an arborescence. If it makes its transition towards \(N_0 = (X_0, \delta_0)\) it is called an anti-arborescence. An arborescence is a subset \(\subseteq V, \mu\) that has one edge out of every node that contains no cycles and has maximum cardinality. For example, if set \(U = \{5, 7, 8, 10\}\) contains 4 elements, then the cardinality of \(|U|\) is 4.

If \(X_i\) and \(X_j\) are the states other than the initial \(X_0\) state, there is a transition from \(X_i\) to \(X_j\), so \(X_j\) has at least one transition. Now, suppose \(X_i\) has two walks, \((X_i, X_{i+1})\) and \((X_j, X_{j+2})\). Concatenating these walks to the walks \((X_{i+1}, X_j)\) and \((X_{j+2}, X_i)\), respectively, we have two distinct changes in state from \(X_i\) to \(X_j\) in \(G_{mc}\). However, in \(\mathcal{K}\), this concatenation is not considered, which makes them Directed Acyclic Graphs (DAG) (see Fig. 2). Most of the biochemical models \(G_{mc}\) can be visualized as DAGs irrespective of the nature of the reactions present in the model. Figure 2 shows the equivalent \(G_{mc}\) tree of shown in Fig. 1. The trees are less complex as they have no cycles, no self-loops. Yet they are still connected, meaning they can depict the state-space.

The weight of the tree containing all \(e\) edges is defined by \(\omega(\mathcal{K}_e) = \prod_{e \in \mathcal{K}_e} \omega(e)\), where \(\omega(e) = \omega(X_i, X_j) = a_{ij}\) is the weight of an edge starting from \(X_i\) and ending at \(X_j\) when \(e \in \mathcal{K}_e\) [36]. For systems which have both forward and backward reactions, if \(n_i\) is the total number of nodes indexed by \(\{1, 2...K\}\) the same as states, then \(n_K\) is the set of nodes carrying \(X_K\), and \(n_0\) is the set of nodes carrying \(X_0\) given \(N_0\) root node of the tree \(\mathcal{K}\), then the walk from one node to another node is given by:

\[
\{f(N_i, N'_i), f(N'_i, N_i) \mid N_0\} \in X_J,
\]

\(\mathcal{K}\) is formed by superimposing the forward transitions between the states \(X_i\) and \(X_j\), with the reverse orientation. Where \(X_j\) and \(X_i\) indicate backward reactions, these are graphically denoted as an individual edge from \(N_i = (X_0, \delta_i)\) to \(N_i = (X_0, \delta_i + 1)\) to \(N_j = (X_0, \delta_i + 2)\) in a tree. The \(N_i\) of \(\delta_i + 2\) can be renamed as a new node. \(N_{i+1}\) remains as it is at a different depth from the \(N_i\) of \(\delta_i\) but contains the same state \(X_i\). In the expansion, repeated states are not considered in the domain; therefore, any node which carries a similar state is considered the same, regardless of the level and indexing. Consideration of trees for the state-space expansion in ISP not only helps to reduce the complexity but also improves the accuracy of the solution of Eq. (5) by identifying nodes which carry probable states. If the Markov chain graph starts in state \(X_i \in X_0\), then the mean number of transits to any state \(X_j\) converging to \(\pi_{X_i, X_j}\) is given by the \((i, j)\)th value of
\[ \bar{X} = \lim_{n \to \infty} \left( \frac{1}{n} \sum_{k=0}^{n-1} A^k \right). \]  

(8)

\( U \) is the set of all arborescences. Let \( U_{X_i, X_j} \) is the set of all arborescences which have a transition from \( X_i \) to \( X_j \) and \( \|U_{X_i, X_j}\| \) is the sum of the weights of the arborescences in \( U_{X_i, X_j} \) then according to the Markov chain tree theorem [33],

\[ \bar{a}_{X_i, X_j} = \frac{\|U_{X_i, X_j}\|}{\|U\|}, \]  

(9)

\( \bar{a}_{X_i, X_j} \) is probabilistic in nature. This nature is not only restricted to the systems which have irreducible Markov chains, in which graph \( G_{mc} \) is strongly connected while carrying probable state-spaces, but also for the systems that can be simplified by converting to a Markov chain tree and then by reducing that tree by ignoring the states which have low probabilities in space \( \phi \).

**Expansion criterion for state space**

As previously mentioned, the states are indexed using \{1, 2, ..., K\} in the domain denoted by set \( X_p \). To derive the time, based on the state-space expansion conditions, the probability exponential form of the CME Eq. (5) is evaluated for approximation up to the desired final time \( t_f \) in steps. To focus on the probable states that contribute most to the probability mass in the domain, we first define the set of non-probable states (those which have the least probability mass) as \( X'_K \), which are to be bunked. The number of states will usually be infinite, without selecting probable states for the domain. By doing this we can avoid recalculating the probabilities and decrease the computational efforts by keeping the domain small. This bunking can also be applied to the

---

**Fig. 2** Equivalent tree of a Markov chain graph, as shown in Fig. 1. This is a special form of graph which has no cycle and no self-loops. It depicts the state-space of the system in the form of a tree (DAG).
initial distribution of the system at $t_0$. If submatrix $A_j'$ contains the non-probable set $X_K'$ of states, then the probability of set will be,

$$P^{(t)}(X_K') = \exp(tA_j')P^{(t)}(X_0). \quad (10)$$

The criterion for defining the non-probable states is determined by the $\tau_m$ tolerance value. $A_j'$ will only be considered to have non-probable states if,

$$A_j' = \begin{cases} 
\text{nonprobable states,} & \text{if } P^{(t)}(X_K') < \tau_m \\
\text{probable states,} & \text{if } P^{(t)}(X_K') \geq \tau_m
\end{cases} \quad (11)$$

Similarly, submatrix $A_j$ has a probable set $X_K$ of states if $P^{(t)}(X_K) \geq \tau_m$ otherwise, the states from $X_K$ are bunked to $X_0'$ if $P^{(t)}(X_K) < \tau_m$. For any iteration, if $P^{(t)}(X_K) \geq \tau_m$ then (from Eq. (11)) some states from $X_K'$ return to $X_K$ in the next iteration to increase the accuracy of the approximate solution ($\mathcal{A}$). The column sum of the approximate solution ($\mathcal{A}$) of these states is defined as:

$$\mathcal{A} = I^T \exp(tfA_j)P^{(t)}(X_0), \quad (12)$$

where, $I = (1,...,1)^T$ is of an appropriate length. Declaring some states as non-probable and removing them before calculating the probabilities as seen in [28] will decrease the accuracy of $\mathcal{A}$ with the cumulative step errors. This can be validated from the state probabilities that have been ignored in the domain:

$$\mathcal{A} = 1 - P^{(t)}(X_K), \quad (13)$$

We define the step error in terms of the probabilities bunked. If $e_{\text{error}} \propto P^{(t)}(X_K')$ then,

$$e_{\text{error}} = 1 - I^T \exp(tfA_j)P^{(t)}(X_0) \quad (14)$$

$$e_{\text{error}} = 1 - \mathcal{A} \quad (15)$$

Every expansion step explores at least one new state and change $\{X_K\}$ but not necessarily $\{X_K'\}$ if:

$$P^{(t)}(X_K) > \tau_m > P^{(t)}(X_K'), \quad (16)$$

is satisfied. For ideal systems with a given initial probability of $P^{(t_0)}(X_0)$, the $\{X_K'\}$ should be null and so $P^{(t)}(X_K') = 0$. For such systems $\{X_K\}, \{X_K\} \in \{X_f\}$ for final projection and,

$$P^{(t_f)}(X_f) = P^{(t_f)}(X_K) + P^{(t_f)}(X_K'), \quad (17)$$

$$P^{(t_f)}(X_f) = P^{(t_f)}(X_K) + 0. \quad (18)$$

$P^{(t_f)}(X_f)$ in Eq. (18) is the solution of Eq. (3) after the state-space is expanded to $X_K$. However, for large biochemical systems, Eq. (18) may not hold completely true, due to the nature (fast ($R_{M(f)}$) and slow ($R_{M(s)}$)) of some reactions present in the system; therefore, the condition in Eq. (11) will pass the states from $X_K'$ to $X_K$. The states with the lowest probabilities will be bunked when:
\[ \mathbb{P}^{(t)}(X_K') < \mathbb{P}^{(t)}(X_K), \]  

(19)

This improves the solution. Removing without calculating the probabilities of some states is one of the lags of the current methods [18, 20, 21, 24, 26–28]; it is a result of achieving the truncated domain and saving computation time. To address this, we set a leakage point based on:

\[ \mathbb{P}^{(t)}(X_K') \geq \tau_{m, \text{leak}}(X_K) > \mathbb{P}^{(t)}(X_0 K/C_{16}/C_{17}); \]

(20)

where, \( \tau_{m, \text{leak}} \) for systems will reform Eq. (16) as:

\[ \mathbb{P}^{(t)}(X_K') \geq \tau_{m, \text{leak}}(X_K) > \mathbb{P}^{(t)}(X_0 K/C_{16}/C_{17}); \]

(21)

which would then zip the \( X_K' \) further by leaking the highest probabilities to \( X_K \) so that the probability sum is no longer conserved. The motivation of setting this ration is to reconsider (up to 40% of \( X_0 K \)) the bunked states to improve the \( \mathbb{P} \) solution and decrease the expansion step error. While modeling the biochemical system, if the slow and fast reaction [12] criterion is considered during expansion, then \( \tau_{m, \text{leak}}(X_K) \) will be defined as:

\[
\tau_{m, \text{leak}} = \begin{cases} 
\tau_m \frac{\text{no. of } R_{M(s)}}{\text{no. of } R_{M(f)}} & \text{if no. of } R_{M(s)} < \text{no. of } R_{M(f)} \\
\text{else,} & \\
\tau_m \frac{\text{no. of } R_{M(f)}}{\text{no. of } R_{M(s)}} & \text{if no. of } R_{M(s)} > \text{no. of } R_{M(f)} \\
\text{else,} & \\
\tau_m * 0.4, & \text{if no. of } R_{M(f)} = \text{no. of } R_{M(s)}. 
\end{cases}
\]  

(22)

We consider Eq. (21) criterion for all the computational experiments in this study. The conditions in Eqs. (21) and (22) will lead to an optimal set of states as,

\[ X_K \leftarrow X_K - X_K'; \]

(23)

at \( t_d \) in the domain. When \( X_K \) is updated at every \( t_{\text{step}} \) before reaching \( t_f \) this creates several intermediate domains which we define as \( \text{Bound} \). At \( t_0 \), the domain only has the initial state of the system; therefore, we define the \( \text{Bound} \) as:

\[ \text{Bound}_{\text{lower}} = \{\text{domain, } \bar{d}_l = 1\} \]

(24)

After a single \( t_{\text{step}} \) of expansion, if \( X_K \) is updated with a new state or set of states, it creates:

\[ \text{Bound}_{\text{upper}} = \{\text{domain, } \bar{d}_l \} \]

(25)

at \( t_d \). Here, \( \bar{d}_l \) denotes the depth level of the latest state or set of states that has been added in the domain to form \( \text{Bound}_{\text{upper}} \). This \( \text{Bound}_{\text{upper}} \) is re-labeled and considered as \( \text{Bound}_{\text{lower}} \) for the next \( t_{\text{step}} \) of the expansion. If the expansion is to be limited in the number of \( \text{Bounds} \), then every \( \text{count}(\bar{d}_{\text{limit}}) \) leads to:

\[ \text{count}(\bar{d}_{\text{limit}}) = \bar{d}_{\text{limit}}, \]

(26)

where, \( \bar{d}_{\text{limit}} \) is the bound limit. For example, if \( \bar{d}_{\text{limit}} = 2 \), then the \( \text{count}(\bar{d}_{\text{limit}}) \) will be from \( 0 \rightarrow 1 \rightarrow 2 \). If the \( \text{count}(\bar{d}_{\text{limit}}) \) is increased to \( \bar{d}_{\text{limit}} \) for \( I_{\text{tr}}^{\text{th}} \) iterations, then \( \text{Bound}_{\text{upper}} \)

\[ \text{Bound}_{\text{upper}} = \{\text{domain, } \bar{d}_{\text{limit}} \} \]

(27)
in the current iteration will be $\text{Bound}_{\text{lower}}$ for the next iteration. Every $\text{Bound}_{\text{lower}}$ state will be the strict subset of every consecutive $\text{Bound}_{\text{upper}}$ given as:

$$\text{Bound}_{\text{lower}}(Z) \subset \text{Bound}_{\text{upper}}(Z).$$

(27)

and the upper bound as:

$$\text{Bound}_{\text{upper}}(Z) = \{\text{Domain at } Z^{th} \text{ iteration}, \bar{A}_i\},$$

(28)

where $Z$ is the number of Bounds (or intermediate domains). The 2D pyramid domain in Fig. 3 graphically illustrates the increase in the population of states in the domain with the increase in iterations ($I_t$). The apex of the pyramid represents the initial state $X_0$ of the system at $\text{Bound}_{\text{lower}}(1)$ at $t_0$ whereas the base represents the deepest level where the system ends with the final domain carrying set $X_K$ with the maximum probability mass.

For large biochemical systems, the number of created Bounds are based on $I_t$. They have million/billions of states. Expansion can be terminated by defining time $t_f$ at which the solution is required. To have an auto break-off point in the expansion, it is first necessary to define the criteria that limits $I_t$, or when no more new states can be searched. Therefore, we define this criterion in the following section. This criterion also applies to biochemical systems which have fast and slow reactions [12].

**Cease of criterion after updating**

In every expansion step, the domain is validated by Eq. (21) and new states are added in $X_K$ as long as:

$$1 - I^T \exp(t_f A_j) \cdot P^{(t)}(X_0) \geq \tau_m,$$

(29)

is satisfied for probable states and stops if Eq. (29) is not satisfied. This leads to a point at $t_f$ where $e_{error} < \tau_m$, but the expansion can be extended to meet accuracy requirements by re-considering the criteria as:

![Fig. 3 General framework of 2D pyramid domain showing increases in domain size concomitant with the increase in state with an increase in the bounds. $\text{Bound}_{\text{lower}}(1)$ represents the initial condition, whereas $\text{Bound}_{\text{upper}}(Z)$ represents the final domain which carries the explored set of states of the system]
1 - IT \exp(t_f A_j) P^{(t)}(X_0) \geq \tau_{m*}^{\text{no. of } R_{M(j)}} \text{no. of } R_{M(i)} \right), \quad (30)

1 - IT \exp(t_f A_j) P^{(t)}(X_0) \geq \tau_{m*}^{\text{no. of } R_{M(i)}} \text{no. of } R_{M(j)} \right), \quad (31)

1 - IT \exp(t_f A_j) P^{(t)}(X_0) \geq \tau_{m*}(\text{leak}) \right), \quad (32)

before steps to $t_f$. However, the size of $X_K$ obtained through Eqs. (30), (31) and (32) at $t_f$ will be greater compared to the size of $X_K$ obtained by Eq. (29) as the latter will have fewer states. In Eqs. (30), (31) and (32), with the increase in the size of $A_j$, the value of the left-hand side will also increase, resulting in an improvement in $Ӕ$. When considering any Markov process of a biochemical system of any size in which the probability density expands according to Eq. (3) then Eqs. (30), (31) and (32) will approximate the solution within $\tau_m^{\text{no. of } R_{M(i)}}$ and $\tau_m^{\text{no. of } R_{M(j)}}$, respectively, of the true solution of the CME, which is Eq. (3).

Computational experimental results

The ISP method is initialized and parameterized using the initial conditions of the models. Due to a large number of mathematical operations and equations, simultaneous parameter predictions with a limited number of experimental values is often complicated for dynamic systems. Therefore, the consistency with the available experimental data was ensured at each step of the ISP. This method has led to the successful development of several functions that integrate large number of processes supporting extensive expansion of the state-space.

To demonstrate the ISP LAS algorithm, we first consider the catalytic reaction system [37] defined by the reactions

$$S \xrightarrow{k_1} B \xrightarrow{k_2} C, \quad B + P \xrightarrow{k_3} B + E \quad (33)$$

depicted as a network in Fig. 4 as:

In this biochemical system (dimension = 5), reactant $P$ will transform into product $E$ via complex $B$ when reactant $S$ acts as a catalyst for the reaction and produces $C$. We rewrite this catalytic reaction system as a network of three reactions:

$$R_1: S \xrightarrow{k_1} B, \quad (34)$$

$$R_2: B \xrightarrow{k_3} C, \quad (35)$$

$$R_3: B + P \xrightarrow{k_5} B + E; \quad (36)$$

with the initial copy counts $S_0= 50$, $P_0= 80$, $B_0 = C_0 = E_0$. The reaction rate parameters are $k_1= 1$, $k_2= 1000$, $k_3= 100$. These species counts are used as a state-space to define the model and these copy counts are tracked as $(x_0, x_1, x_2, x_3, x_4)$. In reaction $R_1$, the copy count of $S$ is reduced by 1, which increases the copy count of $B$ by 1. In reaction $R_2$ the copy count of $B$ is reduced by 1, which increases the copy count of $C$ by 1. In contrast, reaction $R_3$ decreases the $B$ and $P$ counts by 1 and increases the $B$ and $E$ counts by 1. As in $R_3$, $B$ acts as a catalyst to convert $P$ to $E$ and $B$
is retained in the same reaction. We can now define the transitions associated with \( R_1, R_2, R_3 \) in the stoichiometric vector \( V_M \) matrix as:

\[
V_M = \begin{bmatrix}
  v_1 \\
  v_2 \\
  v_3
\end{bmatrix} = \begin{bmatrix}
  -1 & 1 & 0 & 0 & 0 \\
  0 & -1 & 1 & 0 & 0 \\
  0 & 0 & 0 & -1 & 1
\end{bmatrix}.
\] (37)

For \( LAS \) method compatibility, the associated Markov chain of this model is converted into a Markov chain tree with the states in terms of the nodes with additional information such as the number of \( R_M \) reactions required to reach the state. In the growing Markov chain tree, the transition between the nodes:

\[
N_i \xrightarrow{v_e(X_0(t), X_1(t), \ldots, X_K(t))} N_{i+1},
\] (38)

is defined in the typical form of the dictionary \( Dict \). We express the propensity functions of the three reactions in terms of the states \( ([S], [B], [C], [P], [E]) \in \mathbb{N} \). Node \( N_1 = (X_0, \bar{a}_1) \) carries the initial state \( X_0 \) of the system at an initial depth of level 1. Further, \( n_j = (X_0, \bar{a}_1, 2, \ldots) \) is expanded and the states updated by following the \( LAS \) order. The corresponding propensities \( \Delta a_{i,j} \) are updated in the \( A_{i,j} \) matrix in every iteration, based on the \( LAS \) updating trend (for example, see SI 2). The system began with \( S_0= 50, P_0= 80 \). Gradually all the reactants are transformed to products, \( E \) and \( B \). The system ends in \( n_J = (X_1, 2, \ldots, 14666, \bar{a}_l) \).

Figure 5 shows the \( LAS \) method’s response when solved with \( \tau_m = 1e-6 \) for \( t_f = 0.5 \) sec. Due to the nature of the model reaction rates, small steps \( t_{step} = 0.01 \) sec are taken to capture the moments based on non-negative, non-zero states for the domain. \( LAS \) successfully creates the domain of an optimum order, with 14666 states at \( t_f \) by introducing the new states to the domain with time, as shown in Fig (a) in Fig. 5. This pattern demonstrates that the frequency (the number of states at any time \( t \)) of expansion increases in depth when the number of active reactions increase in the system. With the addition of probable states, the domain contains enough probability mass to approximate the solution up to \( t_f \). The states

---

**Fig. 4** Catalytic reaction network with five \( N = 5 \) species \( S, B, C, P, \) and \( E \) in a network defining reaction, as given in Eq. (33)
are updated in sets as seen in Fig (b) of Fig. 5, for the catalytic system after every iteration.

The state-space pattern in Fig (b) of Fig. 5 can be used as a blueprint of the catalytic systems’ state-space to compare with other model’s blueprints for their characteristics and occurrence of reactions. Such a pattern can be used to predict the behavior of large network state-space expansions when the set of occurrences of the initial reactions are similar in different systems. The solution of Eq. (5), up to $t_f$ for the domain created by LAS, is shown in Table 1. The system’s conditional probabilities based on its species are shown in Fig. 6.

In three test runs, the ISP LAS run time for the catalytic system was 4677 secs when solving Eq. (5) with 14666 states. The probability of the species in Figure SI 17 (see SI 9) shows the nature of the reactions affecting each species’ count in the system. The involvement of species $B$ in all the reactions results in its highest probability at $t_f$. Species $B$ also acts as a catalyst for $R_3$, converting species $P$ to $E$; therefore, both have equal probabilities at the time of solution.

Figure 6 shows the total probability bunked at $t'$ while progressing with the expansion. Bunking produces an error (w.r.t approximation), with time when the number of states increases with the expansion. LAS produces minimal error of order $10^{-5}$, as given in Table 1.

To demonstrate the ISP LOLAS algorithm, we consider the coupled enzymatic reactions defined by the reactions

$$S + E_1 \xrightarrow{k_1} C_1 \xrightarrow{k_2} S + E_1, \quad C_1 \xrightarrow{k_3} P + E_1$$

$$P + E_2 \xrightarrow{k_4} C_2 \xrightarrow{k_5} P + E_2, \quad C_2 \xrightarrow{k_6} S + E_2$$

depicted as a network in Fig. 7 as:

This biochemical system (dimension = 6) describes two sets of enzymatic reactions transforming species $S$ into species $P$ and transforming species $P$ back into $S$. We re-write $C$ reactions system as a network of six reactions:
with initial copy counts $S = 50$, $E_1 = 20$, $E_2 = 10$, $C_1 = C_2 = P = 0$ and reaction rate parameters of $k_1 = k_4 = 4$, $k_2 = k_5 = 5$, $k_3 = k_6 = 1$. These species counts are used as a state-space to define the model. These copy counts are tracked as:

$([S], [E_1], [C_1], [P], [E_2], [C_2]) \in \mathbb{N} = (x_0, x_1, x_2, x_3, x_4, x_5)$.

As in the previous example, we can now define the transitions associated with $R_1$, $R_2$, $R_3$, $R_4$, $R_5$, $R_6$ in the stoichiometric vector $V_M$ matrix as:

$$V_M = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} = \begin{bmatrix} -1 & -1 & 1 & 0 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & 1 & 1 & -1 \\ 1 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \quad (42)$$

For the LOLAS method, the associated Markov chain of this model is converted to a Markov chain tree with the states in terms of nodes with additional information, such

![Fig. 6](image-url)
as the number of $R_M$ reactions required to reach the state. In growing Markov chain tree, the transition between the nodes:

$$N_i \xrightarrow{\nu_i(X_0(t),X_1(t),...X_K(t))} N_{i+1},$$

is defined in the typical form of the dictionary $Dict$. We express the propensity functions of the six reactions in terms of the states $(\Sigma, [E_1], [C_1], [P], [E_2], [C_2]) \in \bar{N}$.

Node $N_1 = (X_0, \bar{a}_1)$ carries the initial state $X_0$ of the system at the initial depth (level 1). Then $n_i = (X_k, \bar{a}_i, 2, ...)$ is further expanded and the states updated by following the LOLAS order. The corresponding propensities $\Delta a_{ij}$ are updated in the $A_{ij}$ matrix in every iteration, based on the given LOLAS updation trend (for example, see SI 2). Initially, the system started with $S = 50$, $E_1 = 20$, $E_2 = 10$. Gradually all reactant species were transformed into products resulting in the system ending in $n_J = (X_{1,2}, ... ,8296, \bar{a}_1)$.

Figure 8 shows the LOLAS method response when solved with $\tau_m = 1e-6$ for $t_f = 2.0$ sec. Due to the nature of the model reaction rates, small steps $t_{step} = 0.01$ sec are taken to capture the moments. These are based on non-negative, non-zero states for the domain. LOLAS successfully creates the domain of an optimum order with 8296 states at $t_f$ by introducing the new states to the domain with time, as shown in Fig (a) of Fig. 8. In Fig (b) of Fig. 8, demonstrates that the frequency (the number of states at any time $t$) of expansion increases in depth when the number of active reactions increases in the system. With the addition of probable states, the domain contains enough probability mass to approximate the solution up to $t_f$.

Fig (a) of Fig. 8 depicts state-space expansion which increases the number of additions of new states in the domain. In Fig (b) of Fig. 8, ISP LOLAS unfolds the state-space pattern to update states in the domain and expands 8296 probable states in 2.0 sec. As a blueprint of the dual enzymatic reaction network, the state-space pattern in Fig (b) of Fig. 8 can be compared with other model blueprints in terms of its characteristics and reactions. Such a pattern is considered to predict the behavior of a large network state-space expansion when the set of occurrences of the initial reactions are similar in different systems. The solution of Eq. (5), up to $t_f$ for the LOLAS-created domain is shown in Table 2. The system’s conditional probabilities based on species are shown in Figure SI 18 (see SI 10)
In three test runs, ISP LOLAS run time for the dual enzymatic reaction network was \( \approx 1614 \text{ secs} \) when solving Eq. (5) with 14666 states. The probability of the species in Figure SI 18 (see SI 10) shows the nature of the reactions which affect each species’ count in the system. At \( t_f \) the probabilities of \( E_2 \) and \( C_2 \) remain high compared to \( E_1 \) and \( C_1 \) at different molecular counts. This results in a low probability of \( P \) compared to \( S \). We know that this network transforms species \( S \) into species \( P \) and then transforms species \( P \) back into \( S \). Based on the current probabilities of the species at \( t_f \), the future probability of \( P \) will increase. \( S \) will remain the same or decrease. With this change, the probabilities of \( E_2 \) and \( C_2 \) decrease in comparison to \( E_1 \) and \( C_1 \).

Figure 9 shows the total probability bunched at \( t' \) while progressing with the expansion. The bunking produces an error (w.r.t approximation) with time when the number of states increases with the expansion and provided that, LOLAS produces a minimal error of order, \( 10^{-5} \), as given in Table 2.

We extend the application of our ISP method to simulate a large model of the G1/S network [38] under the condition of DNA-damage. We want to determine the number of states at different points in time and predict the conditional probabilities of the protein species based on events leading to the formation of different complexes in the system.

The G1/S model (dimension = 28) with a DNA-damage signal transduction pathway is considered to be very stiff in nature, so while some molecular counts of certain proteins increase very rapidly others do so slowly. This makes it tough to solve, even for a short time period. The model is solved for \( t_f = 1.5 \text{ sec} \) with \( b_{\text{limit}} = 1 \), \( \tau_m = 10^{-6} \), \( \tau_{\text{step}} = 0.1 \). The systematic exploration of nodes carrying probable states are undertaken in a similar way as discussed in Table SI 4 of SI 3 and depicted (see Figure SI 7 of SI 3) in six stages (denoted as \( \hat{S} \)), representing \( R_M \) reactions with propensity, \( a_{\mu} \), with the arcs as transitions.

The nodes are expanded up to \( t_f \) to enable identification of the reaction channels responsible for variations in the proteins. From the transitioning factor of the 2nd-tier, we

### Table 2 LOLAS’ expansion response and solution at \( t_f \) for the dual enzymatic reaction network

| \( t_f \) (sec) | Run-time (sec) | Domain | Expansion time (sec) | Error at \( t_f \) |
|---------------|----------------|--------|----------------------|-----------------|
| 2.0           | 1614.22        | 8296   | 2.0                  | 5.953e−05       |
| 0.01          | ISP LOLAS      |        |                      |                 |
can see that every node has an average of at least \( \approx 97 \) possible child nodes carrying states. Further, \( \text{Dict} \) is expanded for \( n \)-tiers of the child nodes to add more states to the domain. Additionally, \( n_i = (X_0, \tilde{a}_i, 1, 2, \ldots) \) is expanded and updated, as per the \text{ISP LOLAS} trend (see Table SI 5 of SI 3).

The \text{ISP LOLAS} method response for the number of states in the domain and time, \( t \), is shown in Fig. 10. The initial response suggests that only a few reactions were active until \( t = 0.4 \) sec. After that time, more reactions triggered that explosively take the exploration above 0.5 million states in 0.5 sec. For such a large model, this combination of explosion states was expected as proteins undergo several excursions due to the number of reactions in fractions of time, \( t \). The second explosion of states occurs after 1.0 secs when almost all the reactions (involving the species, given in SI 4.1) become active in the network. The size and colour of the 2D pyramid in Fig (a) of Fig. 10 shows the increase in the size of the domain with the state explosions. The number of sets of states that create the bounds at \( t \) are shown in Fig (b) of Fig. 10. With the exploration of the set of 517584 states, the \( \text{Bound}(3)_{\text{upper}} = \{X_0, 1, 2, \ldots, 604677\} \) is formed at 0.5 sec carrying 604677 states. Some states were bunked at 0.5 secs resulting in approximation errors that reach \( 2.42e^{-06} \) at 0.6 sec. At \( t_f \), the LOLAS ends up with a domain defined by \( \text{Bound}(4)_{\text{upper}} = \{X_0, 1, 2, \ldots, 3409899\} \) carrying 3409899 states with \( 3.52e^{-06} \) approximation errors.

Fig (a) of Fig. 10 demonstrates that the state-space expansion increases the number of additions of new states in the domain. \text{ISP LOLAS} quickly expands the state-space up to \( \approx 3.5 \) million states in 1.5 secs. In Fig (b) of Fig. 10, \text{ISP LOLAS} unfolds the state-space pattern to update states in the domain and expands 3409899 states up to \( t_f \).

The corresponding propensities, \( \Delta a_{i, j} \), are updated in the \( A_{i, j} \) matrix in every iteration, based on the \text{ISP LOLAS} update trend. The system started with the initial state of the protein species and gradually, as protein levels change in the system, it exploits the copy counts that shift the system to a new state. The change in protein levels causes the system to transform into new states: here we see the manifestation of the Markov process of the system. The \text{ISP LOLAS} captures this process and defines several
bounds of the domain at different time intervals, as indicated by the pyramid in Fig. 3. To investigate the expansion of states more closely, the order of bounds at different time intervals, and the number of states present in the bounds, are provided in Table 3. The size of bound created in each duration reveals that for every step, the growth of the domain is eight-to-ten times the previous size of the domain.

The set of nodes $N_1, N_2, \ldots N_{3409900}$ carries unique states representing the set of states $(n_{3409900}) = (X_0, 1, 2, \ldots 3409899)$ that forms the state-space of the model. It is important to note that some proteins are synthesized and promoted by the network itself, as evidenced by some reactions of the pathway, which increase the frequency of the repeated states. However, ISP LOLAS validation does not consider them for the domain.

Equation (5)’s solution, up to $t_f$ for the domain, created by ISP LOLAS, is shown in Table 4.

Over three test runs, the ISP LOLAS’ run time for the G1/S model was 1372 secs for solving Eq. (5), with the optimal domain having 3409899 states. The ISP LOLAS response given in Fig. 11, shows the system’s probabilities bunked at $t'$ during the expansion (w.r.t approximation), when the number of states increases with the expansion,

Table 3  Lower and upper bounds of the domain for the G1/S model given by the ISP LOLAS trend based on bound limit $B_{limit}$

| $Z$ | Bound; $\mathcal{D}_{lower}$ | Bound; $\mathcal{D}_{upper}$ | States | Duration |
|-----|-------------------------------|-------------------------------|--------|----------|
| 1   | $\mathcal{D}_{lower} = \{X_0\}$ formed at $t = 0.0$ sec | $\mathcal{D}_{upper} = \{X_0, 1, 2, \ldots 9808\}$ formed at $t = 0.1$ sec | 9808 | 0.0 – 0.1 sec |
|     | Approximation = 1 | Approximation = 0.999999867 |        |          |
|     | $B_{init} = 1$, count($B_{init}$) = 0, 1 | |        |          |
| 2   | $\mathcal{D}_{lower} = \mathcal{D}_{upper}$ formed at $t = 0.1$ sec | $\mathcal{D}_{upper} = \{X_0, 1, 2, \ldots 87393\}$ formed at $t = 0.2$ sec | 87393 | 0.1 – 0.2 sec |
|     | Approximation = 0.999999847 | Approximation = 0.999999173 |        |          |
|     | $B_{init} = 1$, count($B_{init}$) = 0, 1 | |        |          |
| 3   | $\mathcal{D}_{lower} = \mathcal{D}_{upper}$ formed at $t = 0.4$ sec | $\mathcal{D}_{upper} = \{X_0, 1, 2, \ldots 604677\}$ formed at $t = 0.5$ sec | 604677 | 0.4 – 0.5 sec |
|     | Approximation = 0.999999157 | Approximation = 0.99999701 |        |          |
|     | $B_{init} = 1$, count($B_{init}$) = 0, 1 | |        |          |
| 4   | $\mathcal{D}_{lower} = \mathcal{D}_{upper}$ formed at $t = 1.1$ sec | $\mathcal{D}_{upper} = \{X_0, 1, 2, \ldots 3409899\}$ formed at $t = 1.5$ sec | 3409899 | 1.1 – 1.5 sec |
|     | Approximation = 0.99999699 | Approximation = 0.99999648 |        |          |
|     | $B_{init} = 1$, count($B_{init}$) = 0, 1 | |        |          |
and provided that *ISP LOLAS* produces minimal errors of the order of $10^{-6}$, as given in Table 4 and Fig (a) of Fig. 11. We set the checkpoint to examine the initial state’s probability over time. The response in Fig (a) of Fig. 11 indicates that the probability of the system remaining in the initial (normal) state decreases with time in the presence of DNA damage, which triggers the change in protein levels.

The conditional probabilities of the species’ systems are given in Fig. 14 and SI 8. In the case of DNA-damage, large numbers of the most notable parameters increase, compared to normal conditions (in cell cycle progression). The increase is predominantly related to $x_{14}$ (p21) having a high initial probability, see Fig(14) of Figure SI 19 (see SI 11). The feedback (negative) of $x_{24}$ (p53) increases its probability, see Fig(24) in Figure SI 19 (see SI 11), such as the association rate of $x_{16}$ (p21/CycE/CDK2 – P), the rate of synthesis of $x_{14}$ (p21) by $x_{24}$ (p53), the rate of degradation of $x_{14}$ (p21), and the rate of synthesis of $x_{24}$ (p53) by DNA-damage signal. The conditional probabilities of the two key proteins, $x_{10}$ (p27) and $x_1$ (CycE), are affected by the change in the cell’s response to the level of the DNA-damage signal, see Fig (10) and Fig (3) in Figure SI 19 (see SI 11). The parameters related to $x_{10}$ (p27), as well as $x_1$ (CycE), greatly affect the probability of $x_{21}$ (E2f) with time, see Fig (21) in Figure SI 19 (see SI 11). The impact of $x_1$ (CycE) involves additional parameters related to CycA, because the release of supplementary $x_{21}$ (E2f) depends on $x_{20}$ (Rb – PP/E2f) hyperphosphorylation by the activation $x_7$ (CycE/CDK2 – P), which affects the probability of $x_{21}$ (E2f).

When the release of $x_{21}$ (E2f) is affected, the probability of $x_1$ (CycE) increases, see Fig (3) in Figure SI 19 (see SI 11). This leads to the progression to the S-phase, followed by the temporary suspension of cell cycle progression. The increase in probability of $x_{24}$ (p53) shows cell support to repair the DNA damage. The parameters and the probabilities relating to $x_{14}$ (p21) and $x_{24}$ (p53) become important in the case of DNA damage.

### Table 4

| $t_f = 1.5 \text{ sec}$ | Run-time (sec) | Domain | Expansion time (sec) | Error at $t_f$ |
|-------------------------|----------------|--------|----------------------|---------------|
| ISP LOLAS               | 1372           | 3409899| 1.5                  | 3.52e–06      |

![Fig. 11](image.png)

**Fig. 11** The *ISP LOLAS*’ response for total probability binned at $t_f$ from the domain and checkpoint for examining the initial state probability over time. (a) shows how *ISP* maintains accuracy by keeping low errors. (b) shows the decline in the probability of the system remaining in the initial state in the presence of DNA damage.
damage. When combined, the conditional probability of these parameters indicates the involvement of the DNA-damage signal in the transition of G1/S.

Discussion
In this section, we discuss ISP performance, focusing specifically on the speed and accuracy of the expansion, domain size and accuracy of the solution in comparison with other methods.

Comparison with other methods
An approximation of $10^{-5}$ is used to find the approximate number of realizations required by the SSA for the $10^{-4}$ global error. Realizations were computed until the difference was less than $10^{-4}$ between the known distribution and the empirical distribution.

Approximately $10^6$ and $10^5$ runs were required to obtain the right distribution for the catalytic and dual enzymatic reaction networks, respectively. In the catalytic system, we observe (see Table 5) that both versions of ISP are faster than the OFSP of r-step reachability and the SSA of sliding windows. We attribute this greater efficiency to LOLAS having fewer states and less computational time than the OFSP method. LOLAS has better accuracy at $t_f$. Similarly, the ISP was much faster than the SSA, and the total number of realizations required from the SSA to have an error at $t_f$ still large than that of LOLAS is $10^6$. In the dual enzymatic network, we observe (see Table 5) that both versions of ISP are faster than the OFSP of r-step reachability and the SSA of sliding windows; we attribute the improvement to both ISP variants having an efficient domain with a small approximation error and less computational time than that of the OFSP method and better accuracy at $t_f$. Similarly, both ISP variants were much faster than SSA, as the total number of realizations required to have an empirical distribution with the error at $t_f$ is $\approx 12$ times more than the domain produced by ISP.

We also compared the error at $t_f$ to determine the solution’s efficiency. As seen in the results, the increase in the step error in OFSP affected the solution at $t_f$. Figure 12 (see Fig (a) and (b)), compares the ISP (LAS and LOLAS) with OFSP on the basis of the approximation error at $t$ during the expansion of the catalytic and dual enzymatic

| Table 5 | Comparison of the solution of the catalytic reaction system based on ISP, OFSP and SSA |
|---------|--------------------------------------------------|
| $t_f = 0.5$, $t_{step} = 0.01$ | ISP | OFSP | SSA |
| Catalytic reaction system ($t_f = 0.5$, $t_{step} = 0.01$) | | | |
| Run-time (sec) | 4677 | 2706 | 8767 | 17428 |
| Domain at $t_f$ | 14666 | 13089 | 14665 | $10^6$ Runs |
| Expansion time | 0.5 | 0.5 | 0.5 | - |
| Error at $t_f$ | $1.865e-05$ | $1.532e-05$ | $1.917e-05$ | $\approx 9.81 \times 10^{-3}$ |
| Dual enzymatic reaction system ($t_f = 2.0$, $t_{step} = 0.01$) | | | |
| Run-time (sec) | 2368 | 1614 | 2804 | 6374 |
| Domain at $t_f$ | 8282 | 8296 | 8266 | $10^5$ Runs |
| Expansion time | 2.0 | 2.0 | 2.0 | - |
| Error at $t_f$ | $7.470e-05$ | $5.953e-05$ | $1.065e-04$ | $\approx 9.94 \times 10^{-3}$ |
reaction networks, respectively. Addressing the step error in ISP and the selection of the probable states results in a more efficient solution at $t_f$ compared to OFSP.

The typical firing nature of reactions in the catalytic system makes them stiff. Therefore, the selection of states becomes difficult to approximate. While some species in the system increase abruptly, others do so very slowly because the kinetic parameters ($k_1 = 1$, $k_2 = 1000$, $k_3 = 100$) have large differences: this triggers reactions at different rates. Reaction $R_1$ is categorized as a slow reaction in the network: it affects the fast reaction, $R_2$. As the computation results of Table 5 show, the ISP found that only 13089 probable states were required to solve the system up to $t_f$. This not only saves computational time (see Fig. 13) compared to OFSP and SSA, and improves the solution’s accuracy. In OFSP, applying the compression at every step or after a few steps is still computationally expensive for a model like the catalytic reaction system, as seen in Table 5 and Fig (a) of Fig. 13.

The network shown in Fig. 7 consists of two interlinked enzymatic reaction systems. These systems transform species $S$ and $P$ into each other via the other species, making the system stiff in nature. The selection of states thus becomes difficult for approximation. This is due to some species ($S$ and $E_1$) in the system increasing abruptly, while others take longer to increase. Some of the kinetic parameters ($k_1 = k_4 = 4$, $k_2 = k_5 = 5$) have large differences from other kinetic parameters ($k_3 = k_6 = 1$): this triggers reactions at different rates. Categorized as the fastest reaction in the network, $R$ affects species $S$, $C$, $E$. It is followed by other reactions involving other species. As the computation...
The results in Table 5 show that the ISP LAS indicated that only 8282 probable states are required to solve the system up to $t_f$. Likewise, ISP LOLAS identified that only 8296 probable states are required to solve the system up to $t_f$. This saves computational time (see Fig. 13), compared to OFSP and SSA, and improves the solution’s accuracy. In OFSP, applying the compression at a defined step or after a few steps is still computationally expensive for models like the dual enzymatic reaction system, as seen in Table 5 and Fig (b) of Fig. 13.

The total computation effort required at every step, when compressing the number of states up to $t_f$, is approximately equal to the total computation effort required when the compression is applied in the gaps in some steps on a set of states up to $t_f$. Moreover, the state-space will remain the same at $t_f$, regardless of when the compression is applied.

A comparison of the computational times in Table 5 shows that both versions of ISP are significantly faster than other methods. Figure 14 shows the CPU utilization (%) of LOLAS and OFSP with respect to run-time (minutes). The dedicated throughput (see SI 1.1) between EC2 and EBS was not used to solve the model. The average CPU exertion is about 60%, which is a considerable workload for a given model. The expansion and approximation began when CPU use was at $\approx 1.6422\%$ in the catalytic reaction system and $\approx 1.23\%$ in the dual enzymatic reaction system, at $t=0$ sec. It increases up to 60.0% and then drops down to zero at $t_f$.

### Theoretical interpretation of methods

Although, SSA recognizes the support and wastes no time in searching for the right domain and creating independent realizations which can be run parallel on multi-core environment, solving the system via Eq. (5) is quicker than creating realizations via stochastic simulation [39–41]. This is because the N-term approximation [42, 43] of the probability distribution to create the required number of realizations is always less than, or equal to, the minimal support approximation up to same error. These
realizations were computed until the difference was less than the prescribed approximation between the known distribution and the empirical distribution. In terms of the system dimension, which is usually defined by the number of species in the system, the approximation of Eq. (5) in ISP becomes smaller problem to solve compared to approximation through SSA. This enables ISP to perform better.

In contrast, OFSP creates a hyper-rectangle and applies truncation to guarantee the minimal order domain for the approximation. OFSP truncates the state space after every few steps to ensure the minimum size of the domain and enable greater computational speed. However, differences in reaction firing changes the probability of some states at a later stage; therefore, truncating the state space in OFSP after every few steps or at every step would remove probable states from the domain, which in turn would affect the accuracy of the solution. As a result of this, OFSP’s overall performance is compromised. In contrast, ISP first explores the states based on guided exploration through the BLNP function (see method section (a)) and then leaks the set of states $X^t$ which have the lowest probabilities in the bunker at $t$ without removing them (see Eq. (20)). It recalls these sets of states when the probabilities of these states increase at later time.

In ISP, the time and space complexity (refer to SI 7) of removing and accessing the states in the bunking and recalling process is optimum, compared to the overall time and space complexity of the truncation step in OFSP [20]. As seen in Table 5, the number of states present in the domain for the catalytic reaction network in ISP LAS is approximately equal to number of states present in the domain produced by OFSP. Additionally, the number of probable states in the domain for the dual enzymatic reaction system in ISP LAS is quite more as compared to the domain produced by OFSP. However, better complexity and the guided selection of probable states for the domain produces low approximation errors and means that ISP LAS performs better overall than OFS. Similarly, ISP LOLAS outperforms OFSP in finding the optimum domain due to its bi-directional exploratory nature (see methods section (c)). This feature helps ISP LOLAS to achieve a more accurate solution (see Table 5 and Fig. 12) as well as a quicker computational time (see Fig. 13). These benefits are also due to fact that ISP visualizes the state-space as a Markov chain graph or a tree (see Markov chain as a Markov chain tree section) which ultimately decreases the complexity in the expansion phase.

**Conclusions**

This paper has introduced a novel approach, ISP, to model biochemical systems. This new approach addresses both performance and accuracy problems in CME solutions. Provided all probable states are not added into the domain, up to the desired $t_f$ variants of ISP (LAS and LOLAS) provide systematic ways of expanding the state-space. We have demonstrated the effectiveness of our methods with several experiments using real biological models: the catalytic reaction system, the dual enzymatic reaction system, and the G1/S model (large model). The results and the algorithm’s responses reveal improvements in how different sized biological networks can be modeled: even state-spaces with 3409900 nodes (see Table 3) carrying states up to $\approx$3.5 million can be explored within a reasonable time. The results also show that the domain laid out by ISP had an optimal order and was successful in finding probable system states, all the while maintaining high levels of accuracy and efficient computational timing.
We have compared the ISP results against two popular methods: OFSP (r-step reachability) and SSA (τ leaps adaptive). ISP outperformed the other methods, in computational expense, accuracy and projection size. The ISP was more effective in terms of predicting the behavior of the state-space of the system and in performance management, which is a vital step in modeling large biochemical systems. Unlike other methods, the ISP keeps the lowest states probabilities in the bunker without removing (as removed in OFSP) them, before calculation (as removed without calculation in FSP GORDE). It computes the probabilities at t without computing large numbers of realizations (as done in SSA).

The diverging nature of the ISP response, with respect to OFSP in Fig. 19, also shows that the solution improved with t and at a higher tf. For example, in the large model (case study 2), the computation time was 1372 sec and the solution was 3.52e−06 at tf, which was lower than the small model results (the catalytic reaction system). These results show ISP’s compatibility with the distinct size of biochemical models.

These examples have demonstrated that ISP is a very promising technique for system’s biology. For stiff models, such as the G1/S and Candida albicans models, the ISP yielded plenty of information. Likewise, it provided opportunities for stochastic analysis of large models. ISP can be used to compute the probabilities of the species up to the required time. One could also use ISP to conduct robustness and sensitivity analysis on the dynamics of biochemical systems and to keep track of what reactions are more active in the system at a particular time. ISP is also able to determine the complexity of the system by defining the bounds with number states and keep track of the nested state-space patterns (called the ISP model blueprint) that were updated at the end of each step. Outlining the patterns of expansion of states to predict the projection folds can be used for updating the new states.

We anticipate that the current structure of the ISP variants can be employed for different classes and varieties of biological systems. Additionally, they can be used to compute the configurations with many reactions, as long as the notable part of the state-space density is present between Bound(Z)lower and Bound(Z)upper. When there was a high probability of the molecular population of the species undergoing several excursions in a fraction of time, then the ISP uses a small tstep to capture these moments. While such computations were still challenging in the expansion phase for typical models they can be addressed more closely in combination with the second part of the CME solution: that is, the approximation phase. There are several methods which can be used to address these challenges.

Approximation methods, such as the Krylov sub-space, can be used to effectively compute the matrix exponential times of a vector. While it was mathematically attractive to aggregate the states or decompose the large sparse matrix into a small dense matrix using the Krylov sub-space, this method may not be computationally efficient in the absence of an efficient domain. Performance can also be enhanced by employing the fast math functions, compatible with the multicore environment. We have clearly outlined the core ideas behind the ISP variants. We have highlighted the differences and similarities between them and other methods that cover the computational and theoretical considerations that are essential before any of the approximation methods becomes feasible for an efficient CME solution.
Methods

To understand and predict the dynamics of the state-space response in biochemical systems, we have developed an analytical numerical method called ISP. This method integrates the reactions’ propensities describing the Markovian processes through set of nodes governing set of states of the system. The two variants of ISP, named LAS and LOLAS, consist of several modules that incorporate sets of inputs and functions within several compartments. Figure 15 depicts all the ISP modules. The integrated form is discussed later in Tables 7 and 8.

These modules and sub-modules constitute the ISP method. They track key changes in the components that follow changes in the reaction propensities by population and activation of the species. The modules also describe the dynamics of the biochemical system. The method also permits the time form quantification of state-space based on the size and model dimension.

The ISP states expansion strategy is based on the Artificial Intelligence (AI) standards [44–47], state-space search and relative successor ($S_{ac}$) operator or function which perform operations on inputs. AI refers to the study of intelligent agents [48] of a system that perceives and takes action to successfully achieve goals. Most of the problems can be formulated as searches. They can be solved by reducing to one of searching a graph or a tree. To use this approach, we must specify the successor operator which defines the sequence of actions leading from initial to goal state at different time intervals, that lead to the solution.

In terms of AI, we define the state-space as a set of states in the system we can get to by applying $S_{ac}$ to explore new states of a biochemical network. $S_{ac}$ can be applied explicitly, which maps the current state to the output states, or defined implicitly, in that it acts on the current state and transforms it into a new state. In the state-space graph for biochemical networks, we do not define the goal state (or end state) explicitly. Instead, it is defined by $S_{ac}$ implicitly in intervals based on the nature (fast, slow, reversible and irreversible) of the reactions in the system, the

![Fig. 15 Comprehensive ISP method flow chart. A description of the modules (steps), sub-modules and the list of components are discussed in SI 5](image-url)
duration of expansion and the introduction of stochasticity into the system. This should systematically expand the state-space from $X_K$ at $t$ to $X_{K+1}$ at $t+1$ by going through each node $n_j$ at depth $\delta_1$ of the Markov chain graph to evaluate the Markov processes; the expansion aims to occupy most of the probability mass during $[X_K, X_{K+1}]$, and the Markov processes can be solved for probability distribution at $t + 1$.

Where $X_j$ is the finite set of states and $G_{mc} = (X_j, V_\mu)$ is the Markov chain graph on $X_J$ associated with $A = [a_{ij}]$, given $X_0$ as the initial state and $X_K$ as the set of the explored state, where $X_0 \in X_K$ then the implicit successor is defined as,

$$S_{uc} \rightarrow V_\mu(X_K(t)). \quad (44)$$

Equation (44) gives the new states of the system, where, $V_\mu$ is the set of stoichiometric vectors $v_\mu$ function defining the state transitions from any present state $X_i \in X_K$ to new state $X_j \in X_K$. The sample space in the graph contains the unique state of the system stored in a transition matrix, which satisfies Eq. (7) conditions. This transition matrix is a compressed row format (CSR) [49, 50] based on an index of rows $\rightarrow$ columns delimited by commas generating the dictionary $Dict$ of the model which defines the transitions between nodes in the state-space and the mapping of states. Through $S_{uc}$, we can know nothing more than the neighbors (child nodes) of the current node (states reachable through a single reaction). We then consider these neighbors (child nodes) as our only goal states; there can be many in numbers. In a situation such as this, search trails are referred to as blind or uninformed searches. In the following section, we discuss the infrastructure of an uninformed search, the type of data structure we will be dealing with.

**Infrastructure for searching**

A data structure is required to retain the search track in the graph for problem state-space expansion. For each node, $N_i$, of the tree, we create a structure consisting of five elements:

1. $N_i$.State: represents state $X_i$ in the state-space corresponding to $N_i$;
2. $N_i$.Parent: represents the parent node of the child node $N_i$;
3. $N_i$.Depth: represents the depth of state state $X_i$;
4. $N_i$.Cost: represents the cost $\rho_{N_i,N_{i'}}$ of the transition from $N_i$ to $N_{i'}$ in the state-space;
5. $N_i$.Action: represents the action applied via $S_{uc}$ on the parent node to reach $N_i$.

To explore new states in the system, we consider the initial state $\text{state}(N_1) = (X_0, \delta_1)$ as input to the successor, $S_{uc}$. Once the expansion is initiated, the $Dict$ will temporarily (in run-time) store the information for the transition from one node to another in the state-space that binds to the reaction propensities $a_{ij}$. This shift is denoted by an arrow $\rightarrow$, which shows multiple transitions from the parent nodes to the child nodes containing the end state. The set of nodes $n_j = \{N_1, N_2, ..., N_{S_{N_j}}\}$ is a data structure that incorporates the Markov chain graph $G_{mc}$. We explore all the nodes that store the set
of states $X_K$ as well as some additional information about the state, such as the depth and transition cost, from one state to another in the system. If a set of states $\mathbf{n}_i = X_i$, then $C_{N_iN'_i}$ is the transition cost to reach $state(N'_i) = X'_i$ from $state(N_i) = X_i$ and $depth(n_i) = \delta_i$ defines the depth of the set of nodes in $G_{mc}$, then the standard relation between a set of nodes and a set of states is given by $\mathbf{n}_j = (X_j, \delta_j)_{\mathbf{N}_j}$ or $\mathbf{n}_j = (X_j, \delta_j, C_{N_iN'_i})$ and the standard relation between a single node and a single state is given by $N_i = (X_i, \delta_i)$ or $N_i = (X_i, \delta_i, C_{N_iN'_i})$ if the transition cost is considered.

For example, Fig (a) of Fig. 16 shows the Markov chain graph, $G_{mc}$, with $n_J = 10$, $\delta_i = 4$. Its equivalent tree $K_V$ is shown in Fig (b) of Fig. 16 with $n_J = 15$, $\delta_i = 5$. In the tree nodes $N_i = N_{11} = N_{12}$ carries the same state, $X_1$ at $\delta_i = 1, 2$ and $3$, respectively, where walk $N_2 \rightarrow N_{11}$ and $N_7 \rightarrow N_{12}$ represent the backward reaction of the forward reaction represented by walk $N_1 \rightarrow N_2$ and $N_1 \rightarrow N_7$, respectively.

The set of nodes with states are represented as

$$\mathbf{n}_{1,2,\ldots,10} = (X_{1,2,\ldots,K}, \alpha_{1,2,3,4}) \text{ or } \mathbf{n}_{1,2,\ldots,10} = (X_{1,2,\ldots,K}, \alpha_{1,2,3,4}, C_{N_iN'_i} (\text{min}))$$

In general, the transition cost, $C_{N_iN'_i}$, is defined as:

$$C_{N_iN'_i} = a_{1,2} + a_{2,3} + \cdots + a_{N-1,N}.$$  

($C_{N_iN'_i}$) is the summation of all the propensities $a_\mu$ of the $R_M$ reactions that take the system to its final state. For example, $C_{N_iN_{10}}$ to expand to $state(N_{10}) = X_{10}$ of Fig (a) of Fig. 16 is given by

$$C_{N_iN_{10}} = \begin{cases} \text{Path 1:} & a_{1,2} + a_{2,3} + a_{3,4} + a_{4,6} + a_{6,8} + a_{8,10} \\ \text{Path 2:} & a_{1,2} + a_{2,3} + a_{3,5} + a_{5,6} + a_{6,8} + a_{8,10} \\ \text{Path 3:} & a_{1,2} + a_{2,5} + a_{5,6} + a_{6,8} + a_{8,10} \\ \text{Path 4:} & a_{1,7} + a_{7,8} + a_{8,10} \\ \text{Path 5:} & 0, \text{if not reachable} \end{cases}$$

If these are the possible paths for the expansion that expands $X_K$ at every iteration then $C_{N_iN_{10}} (\text{min})$ will be defined by the only path that has the lowest $P(1)(X'_K)$. This can be generalized as follows:

$$C_{N_iN'_i} (\text{min}) \propto \frac{1}{P(1)(X'_K)}.$$  

which means that in order to minimize the expansion cost for the optimal domain $X_K$ at least one path should have states with high probabilities for $X_K$. It is best to follow the path with $C_{N_iN'_i} (\text{min})$, which leaks the minimum probabilities of the system.
For large biochemical models, there exist infinite cases when the node is unreachable from the initial or another node; such cases are ignored when because some probabilities are always dropped in the approximation. Therefore, as defined by the lowest

\[ P(t)(X^0_K) \]

is strictly limited to,

\[ (49) \]

Upon expanding the root node \( N_1 \), we expand the child nodes carrying new states, and then the child-child nodes are explored. The walk between nodes \( N_i \rightarrow V_{\mu}(X_K^t) \rightarrow N_{i+1} \) is defined by dictionary \( \text{Dict} \). This represents the occurrence of \( R_M \) reactions through \( M \) elementary channels. For Fig (a) of Fig. 16, the typical form of dictionary is given below:

\[ D = ([1\rightarrow2, 7], [2\rightarrow1, 3, 5], [3\rightarrow4, 5], [4\rightarrow6], [5\rightarrow6], [6\rightarrow8], [7\rightarrow1, 8, 9], [8\rightarrow10], [9\rightarrowNil], [10\rightarrow9]), \]

and is indexed with the propensities, \( [a_{ij}] \), for all the \( R_M \) reactions. As the propensities are changing by \( \Delta a_{ij} \), we consider the recent values of \( a_{ij} \) in every iteration of ISP that corresponds to the reactions involved. To make the \( C_{N_iN'_i}(\text{min}) \) feasible for any type of biochemical system (stiff, non-stiff) to capture probable states, it is important to consider the expansion cost for small \( t_{\text{step}} \) (time step). This may be because there are some cases when \( C_{N_iN'_i}(\text{min}) \) to reach two or more different child nodes are equal or very close to each other. In addition, we intend to expand the state-space in the direction of carrying states with high probability mass. To achieve this, we treat or convert our uninform search to an informed search infrastructure at run-time to have intuitive knowledge beyond our reach. Figure 17 shows the limits of our visibility in the state-space.

Consequently, it is important to track the reactions which have high propensity function values. As it is difficult to determine the direction of the expansion, in the following section (a), we develop the post successor function on Bayes’ theorem [31, 32] to prioritize the expansion direction based only on those reactions that can be triggered at
a particular time point. In sections (b) and (c), we outline the direction strategy with the depth and bounds of the expansion.

**Bayesian likelihood node projection function**

Bayesian methods [32, 51] are based on the principle of linking prior probability with posterior probability through Bayes’ theorem [31, 32]. Posterior probability is the improved form of prior probability, via the likelihood of finding factual support for a valid fundamental hypothesis. Therefore, we employ the standards of Bayes’ theorem to develop a function targeted to ensure the quality of the expansion based on \( R_M \) reactions active in the network at any particular moment. For a concise definition for the purpose of fundamentals, refer to SI 6.

To improve the quality of expansion through a projection function, one may find it useful to remove the set of states which have low probabilities before calculating Eq. (3). However, removing these states will compromise accuracy as the step error will increase at every \( t \). Moreover, removing these probabilities will greatly affect the solution, as defined at \( t_f \) (at which a solution is required), for large dimension systems which have large state-spaces, as the step error will be much higher due to dropping probabilities without solving Eq. (3). In large systems, any species may change its behavior after a certain number of firing of reactions triggering inactive reactions in the network that will affect the probabilities of the states. If a change in behavior increases the probabilities of certain states, then removing them in an earlier stage is not wise.

Through the **Bayesian Likelihood Node Projection (BLNP)** function, we seek to predict the posterior probability based on the parent state’s probability and calculate the likelihood of the occurrence of reactions that will take the system from the present state to the future state. Through BLNP, we can capture knowledge about the system that will help us to make better predictions about the future state. We are also able to ensure the accuracy of the solution and an optimal domain.
It is important to decide on the direction of the expansion when choosing the future state of the system, as any reaction can occur and take the system to any new state. To understand this situation more clearly on a node level, we assume a Markov chain graph as shown in Fig. 18 of this system which has almost the same number of species count. In Fig. 19, the expansion is at intermediate position, as the initial state \( \text{state}(N_0) = X_0 \) is already expanded and now the expansion of \( \text{state}(N_0) = X_2 \) can be undertaken. To calculate the likelihood of the occurrence of reactions \( R_1, R_2, R_5 \), we consider the propensities \( a_{i,j} \) as a parameter. \( \Delta a_{i,j} \) depends on the kinetic parameter of the reaction. To assign weight to our belief, we deduce a function that will calculate the probability of reactions occurring and prioritize the expansion in order from reactions resulting in states with high probabilities to reaction giving states with low probabilities. It is important to note that none of the probabilities will be removed before the calculation of Eq. (5). With this function, the likelihood of occurrence of \( R_M \) can be computed.

We consider each node as a junction of the prior reactions \( \{R_1, \ldots, R_M\} \) with propensities \( \{a_{1,N}, \ldots, a_{N,N}^M\} \) having prior likelihood values \( \{b'_{1,N}, \ldots, b'_{N,N}^M\} \) and future reactions \( \{R_1, \ldots, R_M\} \) with propensities \( \{a_{1,N}, \ldots, a_{N,N}\} \) having likelihood values \( \{b_{1,N}, \ldots, b_{N,N}\} \), as given in Fig. 18.

To calculate the likelihood of the reactions, it is necessary to have prior information about the occurrence of reactions. If the expansion is to be done at the initial node \( \text{state}(N_0) = X_0 \) (at level 1), then the prior likelihood value \( b'_{N,N} \) is considered as the initial probability or as =1. Once the initial node has been explored, we can calculate the likelihood of the reactions inductively. To calculate the probabilities \( b_{1,N}, \ldots, b_{N,N} \) of the occurrence of \( R_1, \ldots, R_M \), we first calculate the weighted probabilities \( P_{N,1}(\omega), \ldots, \ldots, P_{N,1}(\omega) \) of a system leaving any state by:

\[
P_{N,1}(\omega), \ldots, P_{N,1}(\omega)
\]

**Fig. 18** Current state \( \text{state}(N_0) = X_0 \) and future states \( \{N_3, N_4, N_5\} = \{X_3, X_4, X_5\} \) with corresponding reactions \( R_1, R_2, R_5 \) and assumed propensities \( a_{1,2} = 38, a_{2,3} = 39, a_{2,5} = 40 \), respectively, at any time \( t \), given \( b_{0,2} = 0.4871, b_{0,3} = 0.5128 \)
and multiply it with the prior probability \( b_{01} \ldots b_{0N} \) of the system. This will calculate the likelihood inductively, as \( R_M \) is responsible for transforming the system to the present state \( \text{state}(N_t) = X_t \) at \( t \), leading to a function,

\[
 b\left(N_{1\ldots NM} | b_{1\ldots N}', N' \right) = \frac{a_\mu (X - v_\mu)}{\sum_{\mu=1}^{M} a_\mu (X - v_\mu)} \cdot b'_{N',N} (X - v_\mu) ' 
\]

where,

\[
 P_{N,1\ldots N',N'}(\omega) = \frac{a_\mu (X - v_\mu)}{\sum_{\mu=1}^{M} a_\mu (X - v_\mu)} ,
\]

\[
 b\left(N_{1\ldots NM} | b_{1\ldots N}', N' \right) = P_{N,1\ldots N',N'}(\omega) \cdot b'_{N',N} (X - v_\mu) ' .
\]

Once \( b(N_{1\ldots NM} | b_{1\ldots N}', N') \) is calculated for all the adjacent nodes, the values are arranged in descending order. Every value is bound to one reaction and represents the likelihood of the occurrence of the reaction that takes the system from the present node to the child nodes. Based on likelihood values (highest to lowest), the corresponding reactions are considered one by one and labelled as true events for expansion. For example, if a system has \( R_1, R_2, R_3 \) reactions that bound to BLNP likelihood values in order from highest to lowest, respectively, then three events take the system to new state. When \( R_1 \) is considered for expansion, \( R_2 \) and \( R_3 \) are labeled false events and \( R_1 \) as the true event. When the second highest BLNP likelihood value is considered, which is for \( R_2 \), then it is labeled the true event and the others, \( R_1, R_3 \), are labeled false events. Similarly, the last and lowest BLNP likelihood value is for \( R_3 \), which is labeled as the true event and the others as false events. All states are added in the domain in order from the 1st true event to the

---

Fig. 19 Node \( N \) as a junction of forward and backward reactions \( R_M \), where \( a'_{1,N}, \ldots a'_{n,N} \) are propensities of the prior reactions. \( b'_{1,N}, \ldots b'_{n,N} \) are the likelihood of the prior reactions.
3rd true event. The Eq. (54) of probabilities $b(N_{N1...NM}|b'_{1...N'-N})$ is what we call a BLNP function.

Figure 18 shows the Markov chain tree for selection present at level 2 (assuming that the initial node is already expanded). Here we calculate the weighted probability of a system leaving state $N_2 = X_2$ by:

$$P_{2,3}(\omega) = \frac{a_\mu(X - v_\mu)}{\sum_{\mu=1}^{N_{0}} a_\mu(X - v_\mu)} = 0.3247$$

similarly, $P_{2,4}(\omega) = 0.3333$ and $P_{2,5}(\omega) = 0.3418$.

At level 2, the conditional probability of the occurrence of reaction $R_1$, given the probability of occurrence of reaction $R_1$ at level 1, is given by:

$$b(N_{2,3}|b'_{0,2}) = \frac{a_\mu(X - v_\mu)}{\sum_{\mu=1}^{N_{0}} a_\mu(X - v_\mu)} \ast b'_{0,2}(x - v_\mu)$$

Similarly, the occurrence of reaction $R_1$ at level 2, given the probability of occurrence of reaction $R_6$ at level 1, is given by:

$$b(N_{2,3}|b'_{6,2}) = \frac{a_\mu(X - v_\mu)}{\sum_{\mu=1}^{N_{0}} a_\mu(X - v_\mu)} \ast b'_{6,2}(x - v_\mu)$$

If at level 1, state($N_1$) = $X_1$ and at level 2, state($N_2$) = $X_2$ are explored through $R_1$ then we say that this is a true event and temporarily consider other events false events with respect to the other reactions. Such a condition holds true for the other two cases, when, at level 1, state($N_1$) = $X_1$ is explored through $R_1$ followed by an exploration of state($N_2$) = $X_2$ either by $R_2$ or $R_5$. Given $b'_{0,2}(X - v_\mu)$ and $b'_{6,2}(X - v_\mu)$, we calculate the likelihood of all the $R_M$ events, as given in Table 6. The likelihood values of future reactions cannot be equal, as they are based on the probabilities of occurrence of prior reactions.

From Fig. 18 and Table 6, we can infer, based on the prior reactions for $R_M$, where $M = \{1, 6\}$ that:

| $b_{N'N}$ | $N_{0,2}$ | $N_{6,2}$ | $b_{N'N}(Value)$ | $R_{next}$ |
|------------|-----------|-----------|----------------|------------|
| $b(N_{2,3}|b'_{0,2})$ | True | False | 0.1581 | $R_{i,1}$ |
| $b(N_{2,3}|b'_{6,2})$ | False | True | 0.1665 | $R_{i,6}$ |
| $b(N_{2,4}|b'_{0,2})$ | True | False | 0.1623 | $R_{i,2}$ |
| $b(N_{2,4}|b'_{6,2})$ | False | True | 0.1709 | $R_{i,6}$ |
| $b(N_{2,5}|b'_{0,2})$ | True | False | 0.1664 | $R_{i,5}$ |
| $b(N_{2,5}|b'_{6,2})$ | False | True | 0.1752 | $R_{i,5}$ |
Case 1 \((R_1)\): At level 2, if the prior reaction is \(R_1\) and holds a true event for \(N_0 \rightarrow N_2\) then:

\[
b(N_{2,5}|b_{0,2}^i) > b(N_{2,4}|b_{0,2}^i) > b(N_{2,3}|b_{0,2}^i)
\]

as per \(b_{N,N'}\) the likelihood of occurrence of reactions will be in the order \(R_5 > R_2 > R_1\).

Case 2 \((R_6)\): At level 2, if the prior reaction is \(R_6\) and holds a true event for \(N_6 \rightarrow N_2\) then

\[
b(N_{2,5}|b_{6,2}^i) > b(N_{2,4}|b_{6,2}^i) > b(N_{2,3}|b_{6,2}^i)
\]

as per \(b_{N,N'}\) the likelihood of occurrence of reactions will be in the order \(R_5 > R_2 > R_1\).

There will be \(M\) number of cases (equal to elementary chemical reaction channels) if there are \(R_M\) prior reactions in the system that bring the system to the current node. The likelihood value will change based on \(b_{N,N'}(X - \nu_t)^'\). The BLNP function cannot be used standalone for expansion because it only assigns weightage to direction for expansion. In the Intelligent state projection section, we have derived the condition for our expansion strategies to work with the Markov chain graph state-space and defined the criteria for the formation of bounds (domain formed at anytime \(t\)) with time. The BLNP function (with expansion strategies), will choose the probable states in large biochemical systems where it is important to capture the moments at time \(t\) that define a system’s behavior. BLNP will be useful for identifying the most active reactions in the system while guiding the expansion towards the set of states with high probability mass.

Latitudinal search strategy

We delve deeper into the first subroutine of the ISP called the Intelligent State Projection Latitudinal Search (ISP LAS). Figure 20 manifests the infrastructure of the LAS strategy, showing \(G_{mc}\), the queue and the domain. LAS’ queue data structure is based on the FIFO (First In, First Out) method. In this method, the oldest state added to the queue is considered first. We define and exploit the direction of expansion step-by-step based on intuitive knowledge (as discussed in section (a)), gained from the probability of future reaction events. We follow the conditions (as discussed in the results section).

![Fig. 20 Infrastructure of the Latitudinal Search strategy, showing \(G_{mc}\), the queue and the domain](image-url)
Furthermore, we show step-by-step how the nodes are explored, and states updated in the domain in $I_r$ iterations.

At level $\tilde{a}_l$, the states are expanded only after all the states at level $\tilde{a}_{l-1}$ have been expanded; that is, the search is undertaken level-by-level and depth $\tilde{a}_l$ increases in every $I_r$ iteration. In the case of networks with reversible reactions, the ISP condition will prevent LAS from returning to the state it came from and also prevent transitions containing cycles resulting in DAG with no repetition of any state whatsoever; however, changes in propensities $a_{i,j}$ are validated. Verifying the explored states in $X_K$ in iterations ensures that the algorithm completes and that a deadlock in the state transition cycles cannot occur.

The time complexity of LAS depends on the average transitioning factor $\Upsilon$ and depth $\tilde{a}_l$ and is given by (see SI 7 for detailed discussion),

$$1 + \Upsilon^1 + \Upsilon^2 + \cdots + \Upsilon^{\tilde{a}_l} + (\Upsilon^{\tilde{a}_l+1} - \Upsilon) = O(\Upsilon^{\tilde{a}_l+1}), \quad (55)$$

where,

$$\Upsilon = \frac{\text{Total no. of walk between different nodes}}{\text{Total no. of nodes explored}} \quad (56)$$

For the nodes at the deepest level $\tilde{a}_l$, all walks are valid except for the very last node which stores the end state of the system. Therefore, once the end state is found, based on Eq. (20), LAS will zip $X_K$, further leaking the highest probabilities to $X_K$ for the solution of Eq. (3) which includes the end state of the system. As no state is ever repeated in the domain, space complexity will decrease when the set of states $X_K$ is bunked at $t$ seconds in iterations if Eqs. (19) and (20) are satisfied. In Eq. (13), $P^{(t)}(X_K)$ is computed according to Eq. (5) (the exponential form of the CME), where $\tau_m$ is the tolerance and $I$ is the identity matrix. Due to this stepping bunking of $X_K$ from $X_K$, the time complexity $O(\Upsilon^{\tilde{a}_l+1})$ reduces to $O(\min(\Upsilon^{\tilde{a}_l+1}, \Upsilon^{|X_j|}))$, where $|X_j|$ is the size of the state-space [13]. In contrast, the expansion of new nodes carrying similar states tend to increase $O(\min(\Upsilon^{\tilde{a}_l+1}, \Upsilon^{|X_j|}))$; however, repetitive states are ignored.

If the input $\tau_m$ is too small, the algorithm automatically uses the default value of $\sqrt{\text{eps}}$. Here $\sqrt{\text{eps}}$ is the square root and $\text{eps}$ is the default value of the epsilon on machine. The expansion of child nodes containing state($N_i$) $= X_j$ stops if the condition of Eq. (32) is not satisfied. If the criterion of slow and fast reaction [12] is considered, then the condition of Eq. (31) or (32) is used, depending on the number of $R_{M(s)}$ and $R_{M(f)}$. Table 7 shows the steps of the LAS method with the embedded BLNP function, from steps 4a to 5b.

LAS will be optimal if the transitions between all the states are uniform; that is, all the $R_M$ reactions have the same propensity values. However, in real biochemical models, this condition is unusual. To see a step-by-step demonstration of the ISP LAS algorithm on a toy model, refer to SI 2. We now turn our attention to the second variant of the ISP. We apply the method to a toy model to see how it differs from LAS.
**ISP**

**Longitudinal-latitudinal search strategy**

**Table 7** Steps of ISP latitudinal search (LAS) algorithm

| Step | Inputs | Algorithm
|------|--------|--------------------------------------------------|
| 0    | Initial node \( N_0, a_i, \nu_a, \tau_{in}, \tau_{up}, t_{step} \) | Initialize: \( Bound_{lower} = X_0, \alpha_0(X_0 - \nu_a) = P^R_0(X_0), A = \emptyset \)
| 1    | Start from parent node \( N_i = (X_0, \alpha)_i \) ← Current State of the system at \( t_0 \) | Flag the current node as explored, update \( A \) and add the state \( X_i \) in the domain so that, if \( 1 - \rho \exp(t, \beta) \).
| 2    | Flag the current node as explored, update \( A \) and add the state \( X_i \) in the domain so that, if \( 1 - \rho \exp(t, \beta) \). | \( P^R(X_0) \geq \tau_{in}(\text{leak}) \) holds true go to Step 3; else stop the algorithm.
| 3    | Sort \( \exp(t, A) \), \( P^R(X_0) \) and shift the set of states in \( X_i \) at \( r \) having smallest probabilities, if \( P^R(X_0) \geq \tau_{in}(\text{leak}) \) and at \( t_0 \) update \( X_i \). | Sort \( b \).
| 4a   | Extend the graph dictionary \( \text{Dict} \) by \( \nu_{i}(X(t)) \) by 1 level to check all the nodes \( n_i = (X_i, \alpha_i, \epsilon_{N_0/N}(\text{min})) \); adjacent to \( N_i \): \( Bound_{upper} \leftarrow R_0(\text{Bound}_{lower}) \) reachable by exactly \( R_0 \) reactions (from fast to slow) having \( \epsilon_{N_0/N}(\text{min}) \). If \( n_k = (X_0, \alpha, \epsilon_{N_0/N}(\text{min})) \) be the set of adjacent nodes such that \( n_k \in n_i \), then go to next Step.
| 4b   | Compute the \( BNP \) function for \( n_k \in Bound_{upper} \). | \( b(N_{in,N_0}(b_{in,N_0}^{b}(X_i - \nu)) \)
| 5a   | If \( n_k = (X_0, \alpha, \epsilon_{N_0/N}(\text{min})) \) \in domain, then update the values of the set of states \( X_i \) present in domain and take domain ← \( \text{domain}_{\text{previous}} \cup \text{domain} \) and go back to Step 1; else if \( n_k = (X_0, \alpha, \epsilon_{N_0/N}(\text{min})) \) \notin domain, then add it to the queue in order, according to reachability and go to the next Step.
| 5b   | Sort \( b(N_{in,N_0}(b_{in,N_0}^{b}(X_i - \nu))) \) in descending order and update queue ← (queue; \( b(N_{in,N_0}(b_{in,N_0}^{b}(X_i - \nu))) \)). | \( \text{queue} \)
| 6    | Pull out the nodes \( n_k = (X_0, \alpha, \epsilon_{N_0/N}(\text{min})) \) from the queue in order and add the set of states \( X_i \) in the domain as \( \text{domain} \leftarrow \text{domain} + X_i \) and take \( \text{domain}_{\text{previous}} \cup \text{domain} \), then go back to Step 1. | \( \text{domain with probable states} \)

---

**Longitudinal-latitudinal search strategy**

Here, we delve deeper into the second sub-routine of ISP called the Intelligent State Projection Longitudinal Latitudinal Search (ISP LOLAS). Figure 21 visually represents the infrastructure of the LOLAS strategy, showing the \( G_{irc} \) stack and the domain. The stack data structure of LOLAS is based on the LIFO (Last In, First Out) method. In this method, the newest state added to the stack is considered first. In particular, we define the bound limit and exploit the direction of the expansion step-by-step based on intuitive knowledge (as discussed in section (a)), gained from the probability of future reaction events and follow the conditions (as discussed in the Results section). Furthermore, we show step-by-step how nodes carrying states are explored in a bidirectional way and how these states were updated in the domain in \( I_{f_i} \) iterations.

The states at level \( \alpha_i \) are expanded only after the neighboring states at level \( \alpha_{i-1} \) have been expanded for \( R_{M_i} \); that is, the search is undertaken level-by-level. Depth \( \alpha_i \) increases...
in the same \( I_{tr} \) iteration up to a certain \( b_{\text{limit}} \) (bound limit). The expansion limit is set by \( b_{\text{limit}} \), \( b_{\text{step}} \) (depth step). In contrast to \( LAS \), it is not set by depth \( d_i \). The \( LOLAS \) search updates the dictionary \( Dict \) of \( G_{\text{reac}} \) by the stoichiometric vector function, \( v_j(X(t)) \) on state at level \( d_i \) to explore the child nodes carrying states on levels \( d_i + 1, d_i + 2 \ldots \ d_i + l \), where \( l = \{1,2,\ldots \infty \} \) and then retracts to level \( d_i \) at which new state exploration decisions can be made. In the case of networks with reversible reactions, the ISP conditions will prevent \( LOLAS \) from returning to the state it came from and prevent transitions containing cycles resulting in \( DAG \) with no repetition of any state whatsoever; however, the change in propensities \( a_{i, j} \) is validated. Verifying the explored states in \( X_K \) in iterations ensures that the algorithm completes and that deadlocks in the state transition cycles cannot occur.

In the absence of \( b_{\text{limit}} \), the algorithm will not retract. It will explore longitudinally by tracking only one \( R_M \) reaction. In addition, the algorithm will not terminate with an optimal order domain carrying a maximum probability mass. This would lead to an increase in the approximation error. Instead, it will terminate when carrying only those set of states as a result of tracking only a few \( R_M \), creating an insufficient domain for approximation. Therefore, by default, the value of \( b_{\text{limit}} \geq 1 \) is kept for large systems and can be increased depending upon the model’s dimension and the availability of the testing environment’s random access memory (RAM). \( LOLAS \)’ worst-case time complexity depends on the average transitioning factor \( T \). Depth \( d_i \) is given by (see SI 7 for a detailed discussion):

\[
(T + 1)T^0 + (T)T^1 + (T - 1)T^2 + \ldots + 3T^{d_i - 2} + 2T^{d_i - 1} + T^{d_i} = O(T^{d_i}).
\]  

(57)

### Table 8: Steps of ISP longitudinal latitudinal search (LOLAS) algorithm

| Step 0 | Inputs: Initial node \( N_0 \), \( b_{\text{step}}, b_{\text{limit}}, a_p, v_p \), \( t_m, t_r, t_{\text{step}} \)  
| Step 1 | Initialize \( \text{count}(N_{\text{start}}) \) and start from parent node \( N_l = (X_o, d) \) ← Current state of the system at \( t_{io} \)  
| Step 2 | Flag the current node as explored, A and add the state \( X_l \) in the domain so that; if \( 1 - \frac{1}{T} \) exp \((t/A)\), \( P^0(X_l) \geq t_{\text{step}}(\text{leak}) \) holds true go to Step 3, else stop the algorithm.  
| Step 3 | Sort expl \((t/A)\), \( P^1(X_l) \) and shift the set of states in \( X_l \) at \( t \) having smallest probabilities, if \( P^0(X_l) \geq t_{\text{step}}(\text{leak}) > P^1(X_l) \) and at \( t_{\text{step}} \) update \( X_l \).  
| Step 4a | For \( b_{\text{step}} \) extend the graph dictionary \( Dict \) by \( v_j(X(t)) \) for \( \text{count}(N_{\text{start}}) \) to check all the nodes \( n_j = X_l(a, d), E_{n_j}(\text{min}) \) adjacent to \( N_l \); \( \text{Bound}_{\text{upper}} \) ← \( R_{\text{step}}(\text{Bound}_{\text{lower}}) \) reachable by exactly \( R_{\text{step}} \) reactions (from fast to slow) having \( E_{n_j}(\text{min}) \). If \( n_j = (X_l(a, d), E_{n_j}(\text{min})) \) be the set of adjacent nodes such that \( n_j \in N_l \), then go to the next Step.  
| Step 4b | Compute the BNP function for \( n_j \in \text{Bound}_{\text{lower}} \)  
| Step 5 | If \( n_j = (X_l(a, d), E_{n_j}(\text{min})) \) in domain, then update the values of the set of states \( X_l \) present in domain and take \( \text{domain}_{\text{previous}} \cup \text{domain} \) and go back to Step 1; else if \( n_j = (X_l(a, d), E_{n_j}(\text{min})) \) in domain, then add it to the stack in order, according to reachability and go to next Step.  
| Step 6 | Pop the top nodes \( n_j = (X_l(a, d), E_{n_j}(\text{min})) \) from the stack and add the set of states \( X_l \) in the domain as \( \text{domain} \) ← \( \text{domain}_{\text{previous}} \cup \text{domain} \) and take domain\(_{\text{previous}} \cup \text{domain} \), and go to next Step.  
| Step 7 | If \( \text{count}(N_{\text{start}}) \) up \( \text{Bound}_{\text{upper}} \) creates \( \text{Bound}_{\text{lower}} \) up to \( \text{Bound}_{\text{lower}} \) then label \( \text{Bound}_{\text{upper}} \) ← \( \text{Bound}_{\text{lower}} \) and go back to Step 1; else if \( \text{count}(N_{\text{start}}) < \text{Bound}_{\text{lower}} \) creates \( \text{domain} \) up to \( \text{count}(N_{\text{start}}) \) then go to next Step.  
| Step 8 | \( \text{count}(N_{\text{start}}) \) ← \( \text{count}(N_{\text{start}}) + 1 \) and go to Step 4a  

**Output**: domain with probable states
LOLAS only stores the transition path to the end state besides the neighbors of each relevant node in the exploration. Once all descendants are updated with the relevant probabilities in the projection, it discards the node from the domain (explored), making it ready for the approximation. LOLAS first considers the $R_1$ reaction and the corresponding stoichiometric vector $v_1$ of the system, to explore all the neighboring states up to bound limit $b_{\text{limit}}$. It then considers $R_2$, $R_3$, ..., $R_M$ for the same $b_{\text{limit}}$ and the corresponding $v_2$, $v_3$, ..., $v_M$ to explore the states. For $\text{count}(b_{\text{limit}})$, LOLAS retracts to the $R_1$ reaction and explores the new neighboring states longitudinally. It then reconsiders $R_2$, $R_3$, ..., $R_M$ to explore the other states in a similar fashion. Provided with this reaction tracking pattern, the $\text{BLNP}$ function alters this trend and guides this tracking by considering reactions in a different order based on their probabilities and the number of probable states of the system.

If the system is ending in a set of state $X_k$ carried by $n_k$ at $t_p$ then LOLAS will explore the states efficiently, as long as $\text{count}(b_{\text{limit}}) \leq b_{\text{limit}}$ otherwise $\text{count}(b_{\text{limit}})$ is reset for further expansion. Choosing the appropriate $b_{\text{limit}}$ and $b_{\text{step}}$ depends on the type of biochemical reaction network and the computing configuration. Starting with a depth $1 \rightarrow b_{\text{limit}}$, $\text{LOLAS}$ explores all the states until they return $null$. It then resets the $\text{count}(b_{\text{limit}})$ and $\text{retracts}$ to explore again. In most cases, fewer states are positioned at the lower level. They increase at a higher level when the number of active $R_M$ reactions increases, so retracting provides the ability to track all the reactions simultaneously. The nature of the $\text{LOLAS}$ expansion means that it is able to find more states at any time $t$ compared to LAS. It is also able to find them at the deepest level of the graph. The states at depth $d_1$ are explored once, the states at depth $d_l - 1$ are explored twice, states at depth $d_l - 2$ are explored three times and so on, until it has explored all the system’s states. If the input $r_m$ is too small, the algorithm automatically uses the default value of $\text{sqr}t(\text{eps})$. Here $\text{sqr}$ is the square root and $\text{eps}$ is the default value of the epsilon on machine. The expansion of the child nodes containing $\text{state}(N_i) = X_i$ stops if the condition of Eq. (32) is not satisfied. If the slow and fast [12] reaction criterion is considered, then either Eq. (31) or (32) conditions are used depending on the number of $R_{M}\text{(up)}$ and $R_{M}\text{(down)}$. Table 8 shows the $\text{LOLAS}$ method, with an embedded $\text{BLNP}$ function from steps 4a to 5b.

Refer to SI 3 for the step-by-step demonstration of the ISP $\text{LOLAS}$ algorithm, where we assume the same toy model system.

Supplementary information
Supplementary information accompanies this paper at https://doi.org/10.1186/s12859-020-03668-2.

| Additional file 1. |

Abbreviations

$a_{ij}$ or $a_{ij}^\dagger$: Propensity of chemical reaction; $\Delta a_{ij}$: Change in propensity; $a_1', a_2', ..., a_{N_i}'$: Propensities of the prior reactions; $a_{ij}^\dagger$: Probability of a jump process from state $X_{i-1}$ to $X_i$ per unit time; $a_{ij}^\dagger$: Probability of a jump process from state $X_{i-1}$ to $X_{i-1}$ per unit time; $A$ or $A_{ij}^\dagger$: Defines the transition between $i, j$ and its weightage; $b_{\text{limit}}$: Exploration bound limit in LOLAS; $b_{\text{limit}}(X_{i-1})$: Prior Bayesian likelihood values of $b_{N_i}, b_{N_i}'$; $b(N_{0}, n_{k})$: Represents Bayesian likelihood value given prior $b_{N_{0}, n_{k}}$; $b_{\text{limit}}$; $b_{\text{limit}}^\dagger$, $b_{\text{limit}}$, $b_{\text{limit}}^\dagger$: Define the set of states $\{X_i, 2, \ldots, 5, 0, 1, 2, \ldots, N_i\}$ at $b_{\text{limit}}$ already present in the domain for current iteration; $b_{\text{limit}}$, $b_{\text{limit}}^\dagger$, $b_{\text{limit}}$, $b_{\text{limit}}^\dagger$: Define the set of states $\{X_i, 2, \ldots, 5, 0, 1, 2, \ldots, N_i\}$ at $b_{\text{limit}}$ already added in the domain at the end of current iteration; $c$, $c_1$, $c_2$: Constants; $C_{N_i, N_i}'$: Total transition walk cost from node $N_i$ to $n_{k}'; \text{Dict}$: Dictionary of the model having transition records; $d$: Exploration depth limit in LAS; $d_{\text{step}}$: Exploration depth step in ISP; $\text{dim}$: Dimension of sub-matrix in Sliding Windows Method; $\text{domain}$: Defines the set of states of domain in current iteration that forms $\text{bound}_{\text{up}}$, $\text{domain}_{\text{previous}}$: Defines the set of states of domain in previous iteration that forms $\text{bound}_{\text{down}}$; $D$: Diagonal matrix whose diagonal entries are one; $e$: Markov chain tree
edge, representing walk from \( N_i \) to \( N_j \); \( e_i \): First unit basis vector in Krylov Subspace Method; \( e_{error} \): Represents error value in calculation; \( \exp \): Exponential function; \( \epsilon_{ps} \): Epsilon; \( \mathcal{T}_e \): Denote the sequence of events \( E_1, E_2, \ldots \). \( f(y) \): Represents the positive real value function of \( y \); \( \Gamma_{error} \): Represents graph associated with the Markov chain tree; \( \mathcal{P}_{error} \)

\[ U_n = \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ u_{21} & u_{22} & \cdots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ u_{n1} & u_{n2} & \cdots & u_{nn} \end{pmatrix} \]

: Upper matrix (Hessenberg Matrix); \( f \): Identity matrix \( I = \text{diag}(1, 1, \ldots \ldots) \); \( \lambda_p \): Denote the iterations in \( \mathbb{S}P \); \( k_{error} \): Kinetic parameter of the chemical reaction where \( M = \{1, 2, \ldots \} \); \( f \): Used as subscript for length of depth, for example \( a_l \);

\[ \eta_n \]: Set of nodes as \( \{ N_1, N_2, \ldots \ldots, N_l \} \); \( \eta_n \): Set of nodes carrying set of \( X_0 \) at any iteration; \( \eta_0 \): Root node carrying initial state \( X_0 \) or \( N_j \); Any node; \( \eta \) or \( \{ S_1, S_2, \ldots, S_N \} \): Number of different species; \( num_{ini}, num_{num} \): Random number generated by uniform random number generator (URN); \( P^\omega(k) (X_0) \)

: Initial probability at \( t = 0 \); \( P^\omega(k) (X_0) \): Probability of set of states at time \( t \); \( P^\omega(k) (\omega) \): Weighted probability of transition from \( N_i \) to \( N_j \); \( R_{error} \): M elementary chemical reaction channels \( \{ R_1, R_2, \ldots \ldots, R_m \} \); \( R_{max} \): M elementary chemical reaction channels of fast reactions; \( \Omega_{error} \): Number of rejections in \( \mathbb{L}^2 \); \( S_n \): Approximate number of states; \( S \): Number of states in expansion \( \{ 1, 2, \ldots \ldots \} \) of the supporting information; \( S_{error} \): Implicit successor or operator; \( sqrt \): Square root; \( t_0 \): Time at which initial conditions of system are defined; \( T \): Time at which \( X_0 \) is dropped from the domain; \( t \): Any random time in seconds; \( t_f \): Time at which \( X_0 \) is updated in the iteration; \( t_f \): Final time at which solution is required; \( T \): Transitioning factor; \( U_{X_0}, \mathcal{T}_e \): Set of all arborescences; \( U \): Define the cardinality of any set; \( \mathbb{v} \): Krylov sub-space method - A column vector of a length equal to the number of different species present in the system; \( \mathbb{v}_0 \) or \( \mathbb{v}_{error} \): Stoichiometric vector represents the change in the molecular population of the chemical species by the occurrence of one \( R_{error} \) reaction. It also defines the transition from state \( X_0 \) to \( X_j \) in the Markov chain tree; \( \mathbb{v}_{error}(X(tD)) \) or \( \mathbb{v}_{error}(X(t)) \): Stoichiometric vector function, where \( X \) is any random state; \( \mathbb{v}_0 \) or \( \mathbb{v}_{error} \): Matrix of all the Stoichiometric vectors \( \{ v_0, v_1, \ldots \ldots, v_m \} \); \( \mathbb{W} \): Probability that is computed inductively by \( \mathbb{W}^{pairs} = \mathbb{W}^{pairs} \) in uniformization method; \( x_1, x_2 , \ldots , x_N \)

: Number of counts of different species \( X \) or \( X_j \) or \( X_0 \): Any random state; \( X_0 \): Initial state or initial condition; \( X_j \): Ordered set of possible states \( \{ X_1, X_2, \ldots \ldots, X_N \} \) of the system; \( X_0 \): Set of new states or domain at any iteration; \( X_0 \): Set of states dropped from domain at \( t \) at any iteration; \( y, y_0 \): Positive integers; \( Y_p \): Poisson process given that \( 0 < y \leq M \); \( Z \): Number of bounds in \( \mathbb{S}P \); \( \mathbb{t}_{error} \): Tolerance value; \( \mathbb{t}_{error}(X_0) \): leakage point; \( \mathbb{A} \): Approximate solution of the CME; \( \omega \): Weight or cost of single transition from \( X_0 \) to \( X_j \). It is equivalent to \( d_{error} X_0 \): Markov chain representing biochemical process; \( \mathbb{R} \): Markov chain tree with \( n_j \) \( \mathbb{X} \): Uniformization rate; \( \mathbb{V}_j \): Number of nonzero elements in \( \mathbb{P}_j \); \( \mathbb{Q} \): Sample space; \( \Omega \): Asymptotic lower bound; \( \mathbb{O} \): Asymptotic upper bound; \( \Theta \): Asymptotic tight bound; \( \{ 1, 2, \ldots, M \} \): Indexing of set of states and set of nodes; \( \mu \): \( \{ 1, 2, \ldots, M \} \): Channels of chemical reaction propensity

Acknowledgements
RK acknowledges the project funding and necessary resourcing received from Lincoln University, New Zealand for the duration of three years.

Authors’ contributions
RK and DK developed the research questions and designed the research. RK developed and implemented the algorithm; DK and SS directed the project, RK and DK wrote the manuscript, which has been read, improved, and approved by all three authors.

Funding
RK acknowledges the writing scholarship received from AgL5 Faculty, Lincoln University, New Zealand to write this paper for a month (NZD2000). Only obligation is to submit a paper to a journal.

Availability of data and materials
All the result data files generated and analyzed during the current study are available in the “isp” repository of https://github.com/kosarwal/isp. Codes are not publicly available due to search engine privacy but are available from the corresponding or first author on reasonable request.

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
Not applicable.

Received: 19 April 2020 Accepted: 17 July 2020
Published online: 11 November 2020

References
1. Roberts RM, Cleland TJ, Gray PC, Ambrosiano JJ. Hidden Markov model for competitive binding and chain elongation. J Phys Chem B. 2004;108(20):6228–32.
39. MacNamara S, Burrage K. Krylov and steady-state techniques for the solution of the chemical master equation for the mitogen-activated protein kinase cascade. Numer Algorithms. 2009;51(3):281–307. https://doi.org/10.1007/s11075-008-9239-y.
40. Jahnke T, Huisinga W. A dynamical low-rank approach to the chemical master equation. Bull Math Biol. 2008;70(8):2283–302. https://doi.org/10.1007/s11538-008-9346-x.
41. Hegland M, Hellander A, Lötstedt P. Sparse grids and hybrid methods for the chemical master equation. BIT Numer Math. 2008;48(2):265–83. https://doi.org/10.1007/s10543-008-0174-z.
42. DeVore RA. Nonlinear approximation. Acta Numer. 1998;7:51–150. https://doi.org/10.1017/S09624929000002816.
43. DeVore RA, Howard R, Micchelli C. Optimal nonlinear approximation. Manuscripta Math. 1989;63(4):469–78. https://doi.org/10.1007/BF01171759.
44. Chijindu EVC. Search in artificial intelligence problem solving. IEEE: African Journal of Computing & ICT. 2012;5(5):37–42.
45. Barr A, Feigenbaum E. The handbook of artificial intelligence vol I. Math Soc Sci. 1983;4:320–4.
46. Korf RE. Artificial intelligence search algorithms. In: Algorithms Theory Comput Handb; 1996.
47. Rudowsky I. Intelligent agents. Commun Assoc Inf Syst. 2004;14(August):275–90.
48. Lawlor OS. In-memory data compression for sparse matrices. In: Proc 3rd Work Irregul Appl Archit Algorithms, vol. 6; 2013. p. 1–6. https://doi.org/10.1145/2535753.2535758.
49. Koza Z, Matyka M, Szkoda S, Mirosław Ł. Compressed multi-row storage format for sparse matrices on graphics processing units; 2012. p. 1–26. https://doi.org/10.1137/120900216.
50. Manoukian EB. Modern concepts and theorems of mathematical statistics. New York: Springer New York; 1986. (Springer Series in Statistics). https://doi.org/10.1007/978-1-4612-4856-9.

Publisher’s Note
Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.