GENERATION OF STATIONARY CONTROL POLICIES
WITH BEST EXPECTED PERFORMANCE FOR A FAMILY OF
MARKOV CHAINS

NOAH BERLOW
Electrical and Computer Engineering Department, Texas Tech University
PO 43102 Lubbock, TX, 79409, USA
noah.berlow@ttu.edu

RANADIP PAL*
Electrical and Computer Engineering Department, Texas Tech University
PO 43102 Lubbock, TX, 79409, USA
ranadip.pal@ttu.edu

Due to the high computational cost associated with calculation of optimal control poli-
cies for a family of Markov chains, sub-optimal policy generation approaches are often
required for development of intervention strategies in systems medicine. In this paper, a
new infinite-horizon suboptimal policy generation algorithm using a sequential-selection
approach with efficient re-calculation of steady-state distributions is presented and com-
pared against brute-force and other sub-optimal algorithms. Our results show that for
a system represented by a family of Markov chains, the presented approach is able to
generate robust stationary control policies with superior expected behavior in a computa-
tionally inexpensive manner.

Keywords: Stationary Control Policy; Genetic Regulatory Network Control; Perturbation
of Markov chains.

1. Introduction
Optimal control of Markov processes has a long history starting from the works of
Richard Bellman\(^1\) and Lev Pontryagin \(^2\). Control of Markov processes has applica-
tions in diverse fields such as finance, operations research, economics, learning
theory and systems medicine. Optimal control approaches have been developed
for finite and infinite horizon control of both completely observable and partially
observable systems given the assumption of knowledge of the underlying Markov
process \(^3\). In real life, we are often faced with the scenario of uncertainty in
estimating the parameters of the underlying Markov process which necessitates de-
velopment of robust dynamic programming approaches. Robust dynamic program-
ning from the perspective of worst case or min-max approach has been recently

\(^*\)Corresponding Author
studied $^4$, $^5$, $^6$. However, the worst case approach is often conservative, giving too much importance to events that have an extremely small chance of occurring. Thus, a Bayesian approach to optimal control design has to be pursued when our objective is to improve the expected chances of success, such as was proposed in $^7$ which showed the advantage of Bayesian instead of min-max design in terms of expected performance. The principle of optimality that is necessary for generation of control policies through a robust dynamic programming (DP) concept $^4$, $^5$ is not valid in the best expected cost control formulation for a family of Markov chains, thus preventing generation of optimal control policies based purely on DP principles. Exhaustive search of stationary policies for Bayesian robustness is almost impossible for a reasonable number of states as the total number of possible policies for $N$ states and $m$ controls is $m^N$. The sub-optimal approach enumerated in $^7$ consisted of generating the optimal control policy for each Markov chain and selecting the one with the minimum cost by evaluating the cost of each policy over the family of Markov chains. For a system containing $L$ Markov chains, this approach has a high computational cost involving calculation of $L$ nominal optimal policies and evaluation of $L^2$ cost functions. Furthermore, the expected cost is minimized from among the individual optimal policies. A policy can exist that is not among the individual optimal policies but can produce a lower expected cost. Thus, to mitigate the issue of restricted control policies and lower the computational cost, we will approach the problem from the perspective of sequential search to generate a control policy and utilize computationally efficient techniques for updating the stationary probability distribution of a Markov chain following a rank one perturbation.

Let us consider the case of binary controls and $L$ possible networks whose dynamical behavior is represented by Markov chains where we want a stationary control policy $\mu$ that will give the best expected cost over the $L$ networks. Since our objective is to improve the steady state probabilities of the networks, we can consider changing one entry of $\mu$ at a time and observe the change in the steady-state probability distributions of the $L$ networks. We then keep the perturbation that produces the lowest expected cost. By extension, we incrementally add additional control actions and remove low performing control act similar to sequential floating forward search approach applied in feature selection literature $^8$. The computational complexity involved in calculation of the $L$ steady state probability distributions for each change will be reduced by using techniques for updating the stationary probability distribution of a Markov chain following a rank one perturbation $^9$. As a single change in $\mu$ corresponds to a change in one row of the transition probability matrix of the network, we will apply techniques from $^9$, $^{10}$ for generating the new steady state probability distribution from the previous steady-state probability distribution with minimal computational cost.

The paper is organized as follows. Section 2 provides the mathematical description of the control problem; Section 3 presents the algorithm for robust control policy generation; the complexity analysis of the algorithm is presented in Section 4;
empirical results are presented in Section 5 and the conclusions are presented in Section 6.

2. Problem Description

To explain infinite-horizon control of Markov processes, let us consider a finite state Markov chain described by the control-dependent one-step transition probability

\[ p_{ij}(u) := P(z_{t+1} = j|z_t = i, u_t = u) \]

where, for all \( t \), the state \( z_t \) is an element of a space \( T \) and the control input \( u_t \) is an element of a space \( C \). When the transition probabilities are exactly known, the states make transitions according to \( \omega := (P^u)_{u \in C} \). In this case, once a control input is chosen, the resulting controlled transition probability matrix is uniquely determined. Let \( \mu = (u_1, u_2, \ldots) \) represent a generic control policy and \( \Pi \) represent the set of all possible \( \mu \)'s, i.e., the set of all possible control policies. Let \( J_{\mu,\omega} \) denote the expected total cost for the average cost per stage infinite-horizon problem between control policy \( \mu \) and transitions \( \omega \):

\[
J_{\mu,\omega}(z_0) = \lim_{M \to \infty} \frac{1}{M} E \{ \sum_{t=0}^{M-1} \tilde{g}(z_t, \mu(z_t), z_{t+1}) \}.
\] (2.1)

where \( z_t \) denotes the state at time \( t \), \( \tilde{g}(z_t, \mu(z_t), z_{t+1}) \) represents the cost of going from state \( z_t \) to \( z_{t+1} \) under the control action \( \mu(z_t) \). \( \tilde{g} \) is higher for undesirable destination states. For the same destination states, \( \tilde{g} \) is higher when the control is active versus when it is not. The control problem here corresponds to minimizing the cost in Eq. 2.1. For ergodic Markov chains, the optimal infinite-horizon average cost is independent of the starting state \( z_0 \). Thus, for ergodic Markov chains the optimal infinite-horizon average cost is independent of \( z_0 \) and is given by:

\[
\Phi(\Pi, \omega) := \min_{\mu \in \Pi} J_{\mu,\omega}.
\] (2.2)

where \( J_{\mu,\omega} = J_{\mu,\omega}(z_0) \forall z_0 \in T \).

To explain the issue of robustness, let us reconsider the nominal control problem described by Eq. 2.2. In case of uncertainties, we can parameterize the class of transitions as \( \Omega := (P^a_u)_{a \in C, u \in F_a} \), where \( F_a \) is the noise parameter distribution.

One of the ways to approach robust intervention in the presence of uncertainty is to consider the worst-case scenario. A minimax(worst-case) intervention policy is defined as a policy whose worst performance over the uncertainty class \( \Omega \) is best among all admissible policies. The minimax robust policy, denoted \( \mu_{mm} \), is the one that satisfies:

\[
\Phi(\Pi, \Omega) := \min_{\mu \in \Pi} \max_{\omega \in \Omega} J_{\mu,\omega}.
\] (2.3)

The worst-case robust policy design approach is typically conservative as it gives higher importance to scenarios which rarely occur in practical scenarios. When our

*Henceforth, we will consider only ergodic Markov chains
objective is to avoid extremely undesirable results, a minimax design is suitable, but when our objective is to improve the expected chances of success, a Bayesian approach is preferable. A Bayesian robust policy minimizes the average cost over the uncertainty class \( \Omega \). Let \( \mu_b \) denote the policy designed to be optimal at point \( b \) of the parameter distribution i.e. \( \mu_b \) minimizes \( J_{\mu,\omega_b} \). Here, \( \omega_b \) refers to the probability transition matrix at point \( b \).

The Bayesian robust policy denoted \( \mu^* \) is the one that satisfies

\[
\Phi(\Pi, \Omega) := \min_{\mu \in \Pi} E_a[J_{\mu,\omega_a}]
\]

(2.4)

where \( E_a \) denotes expectation relative to the parameter distribution.

We will consider the case where the uncertainty class is discrete, the controls are binary, and we have \( L \) possible Markov chains with uniform probability representing the underlying system. The overall system can be represented by two families of related matrices: \( \mathcal{M}^u = \{M^u_1, M^u_2, \ldots, M^u_L\} \) denotes the family of \( L \) probability transition matrices under no control, and \( \mathcal{M}^c = \{M^c_1, M^c_2, \ldots, M^c_L\} \) denotes the family of related matrices under active control. Even though we describe the development in this paper based on binary control actions, the approach can be similarly extended to \( n \)-ary control actions. A stationary control policy on a family of Markov chains \( \mathcal{M}^u \) and \( \mathcal{M}^c \) is a control policy independent of time and dependent on the state of the system. Thus for Markov chains \( \mathcal{M}^u_i \) and \( \mathcal{M}^c_i \) with \( N \) states, the stationary control policy \( \mu \) is a length \( N \) binary vector denoting for each state whether control should be applied (the system transition probabilities will originate from \( \mathcal{M}^c_i \)) or not (the system transition probabilities will originate from \( \mathcal{M}^u_i \)). The decimal representation of the stationary policy \( \mu \) will have a range from 0 to \( 2^N - 1 \). Let \( \mathcal{M}^\mu = \{M^\mu_1, M^\mu_2, \ldots, M^\mu_L\} \) denote the controlled set of Markov chains for control policy \( \mu \). For each matrix \( \mathcal{M}^\mu_j \), \( j \in [1, \ldots, L] \), the \( i \)th row of \( \mathcal{M}^\mu_j \) is defined as

\[
\mathcal{M}^\mu_{ji} = \begin{cases} 
M^u_{ji} & \text{if } \mu_i = 0 \\
M^c_{ji} & \text{if } \mu_i = 1
\end{cases}
\]

(2.5)

Note that for fixed transition probabilities, dynamic programming type approaches can be applied to minimize Eq. 2.2 as the resulting system would satisfy the principle of optimality. To illustrate the non-validity of the Principle of Optimality for the Bayesian case, let us consider the definition of the principle of optimality: \(^1\) An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision. If the principle of optimality is valid for this family of networks, then the Bayesian optimal control policy should be equivalent to the optimal policy for the single Markov chain \( 1/L \sum_{i=1}^L \mathcal{M}^u_i \). The principle of optimality will entail that individual step optimization should result in a final optimal policy. We will prove that this is not
applicable by a contradicting example. Let us consider a two-step optimal control problem for a family consisting of two Markov chains i.e. \( L = 2 \) where terminal cost at time step 2 is \( C_2 = [1\ 0]^T \) and \( \mathcal{M}_1^* = [0.7298 \ 0.2702; 0.1871 \ 0.8129] \), \( \mathcal{M}_2^* = [0.1724 \ 0.8276; 0.5334 \ 0.4666] \), \( \mathcal{M}_1^* = [0.2293 \ 0.7707; 0.3308 \ 0.6692] \), \( \mathcal{M}_2^* = [0.6693 \ 0.3307; 0.8962 \ 0.1038] \). Note that for applying dynamic programming in this scenario, we will start at time step 2 and trace backward to time step 1 and 0. The optimal one step cost at time step 1 for state 1 is given by \( \min(0.5 \ast (\mathcal{M}_1^* + \mathcal{M}_2^*) \ast C_2, 0.5 \ast (\mathcal{M}_1^* + \mathcal{M}_2^*) \ast C_2) = 0.4493 \). Similarly, the optimal one step cost for state 2 is 0.3603. If the principle of optimality is valid in this scenario, then the optimal cost at time step 0 can be calculated using the new cost vector \( C_1 = [0.4993 \ 0.3603]^T \). Thus, the optimal 2 step cost for state 1 should be given by \( \min(0.5 \ast (\mathcal{M}_1^* + \mathcal{M}_2^*) \ast C_1, 0.5 \ast (\mathcal{M}_1^* + \mathcal{M}_2^*) \ast C_1) = 0.4003 \). Similar calculations will produce the optimal 2 step cost for state 2 as 0.3923. The corresponding control actions are \( \mu_1 = [1\ 0] \) at time step 1 and \( \mu_0 = [1\ 0] \) at time step 0. However, if we apply the control \( \mu_1 = [0\ 0] \) at time step 1 and \( \mu_0 = [1\ 1] \) at time step 0, then the final 2 step control cost is 0.3016 for state 1 and 0.2882 for state 2, which are lower than the costs arrived by assuming the validity of the principle of optimality (0.4003 and 0.3923). This example illustrates that the principle of optimality is not valid for Bayesian scenarios. Note that when the control \( \mu_2 = [0\ 0] \) is applied at time step 1, the corresponding costs are [0.4511 0.3603] and thus the cost 0.4511 for state 1 is higher than the one step optimal cost of 0.4493. Thus, this policy does not produce the optimal 1 step cost but produces better results in 2 time steps.

A basic approach for Bayesian control policy generation can be evaluating all possible control policies and selecting the one that produces the minimum cost, but this brute-force approach will have enormous complexity. A sub-optimal approach can be generating the \( L \) optimal policies for each individual Markov chain pair, \( \{\mathcal{M}_i^*, \mathcal{M}_i^*\} \) for \( i \in \{1, \cdots, L\} \), evaluating the expected cost \( E_a[J_{\mu,\omega}] \) for each individual policy and selecting the one with the minimum cost. This approach was applied in 7 and we will term it \( L_{\text{min}} \) approach. Note that the Bayesian robust policy can be different from the individual optimal policies; as such, we will consider a sequential approach where at each stage, a control action for a specific state that minimizes the average control cost is selected. The computational complexity involved in the evaluation of \( E_a[J_{\mu,\omega}] \) for change in one dimension of \( \mu \) will be minimized by using perturbation theory of Markov chains for rank one perturbations.

3. Algorithm for Stationary Control Policy Calculation

If we revisit Eq. 2.1 and consider the case where \( g(z_t, \mu_t, z_{t+1}) \) is only dependent on the final state \( z_{t+1} \), the Markov chain is ergodic, and the cost of control is zero, then minimizing \( J_{\mu,\omega} \) will be equivalent to maximizing the steady state probabilities of desirable states and minimizing the steady state probabilities of undesirable states. In fact, if \( G \) is a length \( N \) vector representing the cost of the \( N \) states and
\(\pi_i^\mu\) represents the steady state probability distribution of \(M_i^\mu\) for \(i \in [1, 2, \cdots, L]\), then \(E_{i\nu}[M_{i\nu}] = \sum_{i=1}^{L} G \cdot (\pi_i^\mu)^T\). We will denote \(E_{i\nu}[M_{i\nu}]\) by \(\Gamma(\mu)\) henceforth. We next present the algorithm for generation of stationary control policies based on a sequential floating search approach similar to sequential approaches in feature selection\(^8\). For the following algorithm, \(\mu\) will be denoted by a set \(S\) containing the indices \(i\) for which the control is on i.e. \(\mu_i = 1\) and set \(F\) will denote application of control for all states i.e. \(F = \{1, 2, \cdots, N\}\). Note that the cost of control is considered to be zero in the above formulation. If we want to incorporate the cost of control, then \(\Gamma(\mu) = \sum_{i=1}^{L} (G + f(\mu)) \cdot (\pi_i^\mu)^T\) where \(f(\mu)\) is a \(N\) length vector denoting the cost of control. For example, if cost of control is 1 when control is applied and 0 when control is not applied, then the \(i\) entry of \(f(\mu)\) is 1 if control is applied at state \(i\) and 0 if no control is applied for state \(i\).

### 3.1. Sequential Algorithm for Control Policy Generation Utilizing Steady-State Distribution Recalculation

The proposed algorithm begins with a stationary control policy of no control for all the states. Then it searches for the state \(x_1\) that should be controlled for maximum cost reduction. Let us assume that the cost following application of control at \(x_1\) is \(\Phi(1)\). Once the state with maximum reduction in cost is selected (i.e. \(x_1\)), the next state \(x_2\) that results in the greatest reduction of cost in conjunction with \(x_1\) is found. Thus \(x_2 = \arg\min_{x \in F \setminus x_1} \{\Gamma(x_1 \cup x)\}\). At any stage \(k\) of iteration of the algorithm, \(x_1, x_2, \cdots, x_k\) are selected for application of control and \(\Phi(i)\) for \(i = 0, 1, \cdots, k\) denotes the smallest cost that has been achieved by the algorithm with application of control on \(i\) states. Next, we test if removing any one of the \(k\) states produces a cost that is lower than our calculated cost at the \(k-1\) stage \(\Phi(k-1)\); if one is found then that state is removed from the control policy. As an example, if we have selected in order the states \(x_1, x_2, x_3, x_4, x_5\) then we will search if one of the following sets of states \(\{x_1, x_2, x_3, x_5\}\), \(\{x_1, x_2, x_4, x_5\}\), \(\{x_1, x_3, x_4, x_5\}\), \(\{x_2, x_3, x_4, x_5\}\) produces a cost lower than the best cost at stage 4, \(\Phi(4)\). If, for example, \(\{x_1, x_3, x_4, x_5\}\) produces a cost lower than \(\{x_1, x_2, x_3, x_4\}\), then \(x_2\) will be removed, the best 4 states for control will be \(\{x_1, x_3, x_4, x_5\}\), and \(\Phi(4)\) will be updated with the cost of \(\{x_1, x_3, x_4, x_5\}\). This backward step is a way to consider state combinations which have not been explored. The algorithm for generation of stationary control policies based on the described sequential logic is presented in Algorithm 3.1. To mitigate the computational cost incurred by recalculation of the steady state distributions of \(M^\mu\), we utilize the fact that altering a control policy for a single state is equivalent to a rank one perturbation of a Markov chain, as only one row of the Markov chain is altered. Based on perturbation theory of Markov chains\(^9\), the steady state distribution of a Markov chain following a rank one perturbation can be calculated from the prior steady state distribution with reduced complexity of \(O(N^2)\). The algorithm for efficient calculation of the system cost \(\Gamma\) is presented in section 3.2.
Algorithm 3.1 Algorithm for generating stationary control policy

Require: \( G \): State Cost Vector
\[
M^u = \{ M^u_1, M^u_2, \ldots, M^u_L \}
\]
\[
M^c = \{ M^c_1, M^c_2, \ldots, M^c_L \}
\]
Ensure: \( S \), the suboptimal stationary control policy

\[
S = \emptyset
\]
\[
\Phi = \text{zeros}(N + 1)
\]
\[
count = 0
\]
\[
\Phi(count) = \Gamma(S)
\]
\[
x_1 = \arg\min_{x \in F \setminus S} \{ \Gamma(S \cup \{ x \}) \}
\]
while \( \Gamma(S \cup \{ x_1 \}) < \Gamma(S) \) and \( |S| < N \) do
\[
S = S \cup \{ x_1 \}
\]
\[
count = count + 1
\]
\[
\Phi(count) = \Gamma(S)
\]
\[
x_2 = \arg\min_{x \in S} \{ \Gamma(S \setminus \{ x \}) \}
\]
while \( \Gamma(S \setminus \{ x_2 \}) < \Phi(count - 1) \) do
\[
S = S \setminus \{ x_2 \}
\]
\[
\Phi(count - 1) = \Gamma(S)
\]
\[
count = count - 1
\]
\[
x_2 = \arg\min_{x \in S} \{ \Gamma(S \setminus \{ x \}) \}
\]
end while
\[
x_1 = \arg\min_{x \in F \setminus S} \{ \Gamma(S \cup \{ x \}) \}
\]
end while

3.2. Applying Perturbation Theory of Markov chains to calculate \( \Gamma \)

Consider the rank one perturbation of row \( r \) of a Markov chain \( P^{(1)} \) with steady-state distribution \( \pi^{(1)} = (\pi^{(1)}_1, \pi^{(1)}_2, \ldots, \pi^{(1)}_N) \) resulting in a new Markov chain \( P^{(2)} \) represented by

\[
P^{(2)} = P^{(1)} + a^T \cdot b
\]  \hspace{1cm} (3.1)

where \( a \) and \( b \) are \( 1 \times N \) row vectors, with \( b \cdot e^T \neq 0 \), where \( e \) is the row vector \([1, 1, \cdots, 1]\). Since \( P^{(2)} \) and \( P^{(1)} \) differ only in row \( r \), this implies \( a = e_r \), the \( 1 \times N \) row vector which is 1 at position \( r \) and 0 elsewhere. Additionally, since \( b \) represents the difference between \( P^{(2)} \) and \( P^{(1)} \)

\[
b = P^{(2)}_r - P^{(1)}_r
\]  \hspace{1cm} (3.2)

where \( P^{(i)}_r \) for \( i = 1, 2 \) denotes the \( r \)th row of transition probability matrix \( P^{(i)} \).

The fundamental matrix \( Z^{(1)} \) for an irreducible Markov chain \( P^{(1)} \) with steady state distribution \( \pi^{(1)} \) is given by the equation

\[
Z^{(1)} = [I - P^{(1)} + e^T \cdot \pi^{(1)}]^{-1}
\]  \hspace{1cm} (3.3)
For Markov chain $P^{(1)}$ with row $r$ being perturbed, let

$$\beta = (\beta(1), \beta(2), \ldots, \beta(N)) = (P^{(2)} - P^{(1)})[I - P^{(1)} + e^T \cdot \pi^{(1)}]^{-1} = b \cdot Z$$

(3.4)

Using $\beta$, the steady-state distribution of $P^{(2)}$, denoted by $\pi^{(2)} = (\pi^{(2)}(1), \pi^{(2)}(2), \ldots, \pi^{(2)}(N))$, can be calculated directly. For each $\pi^{(2)}(i) \in \pi^{(2)}$,

$$\pi^{(2)}(i) = \pi^{(1)}(i) + \pi^{(1)}(r) [\beta(i)/(1 - \beta(r))]$$

(3.5)

In matrix form, this can be calculated quickly based on the following equation

$$\pi^{(2)} = \pi^{(1)} + \pi^{(1)}(r) [\beta/(1 - \beta(r))]$$

(3.6)

To allow for further changes to the control policy a new fundamental matrix needs to be constructed for the matrix resulting from the perturbation, $P^{(2)}$. The new fundamental matrix, $Z^{(2)}$, can be calculated directly from $Z^{(1)}$ without the need for a matrix inversion operation. Again, suppose $P^{(1)}$ has undergone a perturbation in row $r$. Let all notations be as previously described. $Z^{(2)}$ can be calculated as:

$$Z^{(2)} = \left[ I - \frac{(\pi^{(1)} \cdot e^T \cdot e^T \cdot b \cdot Z^{(1)})}{1 - b \cdot Z^{(1)} \cdot e^T} \right] \left[ Z^{(1)} + \frac{Z^{(1)} \cdot e^T \cdot b \cdot Z^{(1)}}{1 - b \cdot Z^{(1)} \cdot e^T} \right]$$

(3.7)

Finally, the algorithm for generating $\Gamma(\mu)$ using the described approach is presented in Algorithm 3.2.

### Algorithm 3.2 Algorithm for generating $\Gamma(\mu)$ for change in the $r$th row

**Require:**
- $G$: State Cost Vector
- $r$th row perturbations: $b_1, b_2, \ldots, b_L$
- Fundamental Matrices: $Z^{(1)}_1, Z^{(1)}_2, \ldots, Z^{(1)}_L$
- Steady State Distributions: $\pi^{(1)}_1, \pi^{(1)}_2, \ldots, \pi^{(1)}_L$

**Ensure:**
- $\Gamma(\mu)$, the expected cost of the stationary policy $\mu$
- $Z^{(2)}_1, Z^{(2)}_2, \ldots, Z^{(2)}_L$, the new fundamental matrices
- $\pi^{(2)}_1, \pi^{(2)}_2, \ldots, \pi^{(2)}_N$, the new steady state distributions

**for** $i = 1$ to $L$

$$\beta_i = b_i Z^{(1)}_i$$

$$\pi^{(2)}_i = \pi^{(1)}_i + \pi^{(1)}(r)[\beta_i/(1 - \beta_i(r))]$$

$$Z^{(2)}_i = \left[ I - \frac{(\pi^{(1)}_i \cdot e^T \cdot e^T \cdot b_i \cdot Z^{(1)}_i)}{1 - b_i \cdot Z^{(1)}_i \cdot e^T} \right] \left[ Z^{(1)}_i + \frac{Z^{(1)}_i \cdot e^T \cdot b_i \cdot Z^{(1)}_i}{1 - b_i \cdot Z^{(1)}_i \cdot e^T} \right]$$

**end for**

$\Gamma(\mu) = 0$

**for** $i = 1$ to $L$

$$\Gamma(\mu) = \Gamma(\mu) + \pi^{(2)}_i \cdot G^T$$

**end for**
4. Complexity Analysis

Let \( \mu = [0, 0, 0, \cdots, 0] \) denote the initial stationary control policy. For utilizing the sequential search algorithm, the necessary first step consists of calculating the steady-state distribution of the matrices in family \( \mathcal{M}^\mu \). The steady-state probabilities of an \( N \times N \) matrix can be computed by a matrix inversion with a complexity of \( O(N^3) \) using Gaussian Elimination. Thus, the initial step will have a complexity of \( O(LN^3) \) computation.

The next step is computing the \( Z^{(1)} \) matrices, as each \( Z^{(1)} \) matrix is a necessary element in the computation of further steady-state probability distributions. Since, there are \( L \) \( Z^{(1)} \) matrices associated with the family of Markov chains, the computational cost of computing the \( L \) fundamental matrices is \( O(LN^3) \) assuming complexity of \( O(N^3) \) for the inverse operation and \( O(N^2) \) for the matrix addition, subtraction and column multiplication.

At this stage, the necessary prerequisites for the algorithm have been met. The incremental improvement of the stationary control policy \( \mu \) begins with recalculation of the steady-state probability distributions affected by rank one perturbations. Since the greatest improvement can occur under any possible added control, it is necessary to calculate the effect of every rank one perturbation on every matrix in the family \( \mathcal{M}^\mu \). Equation 3.6 provides the method for calculating \( \pi^{(2)} \), the new steady state distribution. This calculation requires \( N^2 + 2 \cdot N + 1 \) operations for a single steady-state recalculation. This calculation must be performed on every matrix in \( \mathcal{M}^\mu \) and for each row \( j \) for each \( M^\mu_{ij} \in \mathcal{M}^\mu \), resulting in \( N \cdot L \cdot (N^2 + 2 \cdot N + 1) \) operations to compute all possible perturbed steady-state probability distributions. For subsequent additions to the stationary control policy, the number of rows requiring perturbation calculations decreases directly with the size of the stationary control policy: if \( f(\mu) \) denotes the number of active controls in \( \mu \) and \( f(\mu) = m \), the cost of calculating the \( m + 1 \)th element of \( \mu \) is reduced to \( (N - m) \cdot L \cdot (N^2 + 2 \cdot N + 1) \). The other steps for calculating the cost of the control policy based on the steady state probability distributions requires additions and comparisons and thus these steps have complexity significantly lower than calculating the steady state probability distributions.

After two iterations, the last step of the algorithm comes into play: the floating or backtracking, portion of the search. The behavior of the backward search mimics that of the forward in all respects except the backward only calculates new steady-state probability distributions for states already in the stationary control policy. Thus, if \( f(\mu) = m \), the reverse step requires \( (m - 1) \cdot L \cdot (N^2 + 2 \cdot N + 1) \) calculations and \( m \) comparisons to determine the control, if any, to remove from \( \mu \). Suppose the algorithm, in the process of finding the suboptimal control policy, makes \( e_\alpha \) element additions and \( e_\beta \) element removals. In practice, \( e_\alpha \) would be usually of the same order as \( N \). The algorithm then requires approximately \( L \cdot N^3 + e_\alpha \cdot N \).

---

\(^b\) An improved complexity of \( O(N^{2.376}) \) can be achieved by Coppersmith and Winograd method.\(^{13}\)
Without the floating part, the presented algorithm has a worst case computational complexity of $O(L \cdot N^4)$. For the floating part, when the number of backwards steps at each step are low (around 2 or 3), the complexity of the presented approach is still $O(L \cdot N^4)$. In comparison, the brute force algorithm has a much larger computational complexity of $O(L \cdot N^3 \cdot 2^N)$.

5. Empirical Performance of the proposed algorithm

In this section, we present simulation results comparing the proposed algorithm with exhaustive search and a competing sub-optimal algorithm. Additionally, a scenario from systems medicine is considered to illustrate the performance of the proposed algorithm. Genetic Regulatory Networks are often represented as Markov chains, such as Stochastic Master Equation models that are a form of continuous time Markov chains \(^{14, 15}\) or Probabilistic Boolean Network models which are coarse-scale Markov chain models \(^{16}\). One of the objectives of Genetic Regulatory Network (GRN) modeling is to design and analyze therapeutic intervention strategies aimed at moving the network out of undesirable states, such as those associated with disease, and into desirable ones \(^{11}\). However, limited experimental data prevents accurate inference of the mathematical model of the GRN. Often, the inverse problem of inference of a GRN model from experimental data can result in a family of possible models instead of a unique model. In this section, the problem considered will be generation of an intervention policy with best expected performance for a GRN represented by a family of Markov chains. The performance is measured by the ability to increase the steady state probability of desirable states following application of control. The desirability of a state is dependent on the state of molecular markers representative of diseases. For instance, a high expression for proteins K-Ras and ERBB2 and inactivation of p-53 in non-small cell lung cancer \(^{17}\) can denote proliferation. In such a scenario, undesirable states will be states with high K-Ras and ERBB2 and low p-53 whereas desirable states will be low K-Ras, low ERBB2 and high p-53. The expected cost to be minimized is as shown in Eq. 2.4.

5.1. Comparison against Exhaustive Search Methods

To analyze the empirical performance of our algorithm, we considered various sets of coarse-scale Markov chains representing genetic regulatory networks. We considered $n$ genes with binary states (thus $N = 2^n$); $L = 1, \cdots, 5$ and the family of Markov chains are generated from an initial random Markov chain using the relationship described in \(^7\). The control problem formulation is as described in Eq. 2.4 and the desirable states are considered to be the states where expression of gene 1 is low (binary zero). We present the results for the case of $N = 2^4 = 16$ (where brute-force cost remains reasonable) in Table 1. $B_{\text{cost}}$ denotes cost reduction by brute-force optimal policy, $S_{\text{cost}}$ denotes cost reduction by the proposed sub-optimal algorithm,
$B_{time}$ and $S_{time}$ represents time taken in seconds for 10 runs by the brute force and proposed sub-optimal algorithms respectively. From Table 1, we should note that the stationary control policy generated by the proposed sub-optimal algorithm has the same performance as the brute force algorithm but requires significantly lower computational time. The proposed approach cannot guarantee optimal rewards for every instance, but the simulation results illustrate close performance.

<table>
<thead>
<tr>
<th>L</th>
<th>$B_{cost}$</th>
<th>$S_{cost}$</th>
<th>$B_{time}$ (s)</th>
<th>$S_{time}$ (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>.08</td>
<td>.08</td>
<td>93.3</td>
<td>.76</td>
</tr>
<tr>
<td>10</td>
<td>.23</td>
<td>.23</td>
<td>147.8</td>
<td>2.3</td>
</tr>
<tr>
<td>15</td>
<td>.14</td>
<td>.14</td>
<td>209.5</td>
<td>2.5</td>
</tr>
<tr>
<td>20</td>
<td>.04</td>
<td>.04</td>
<td>123.1</td>
<td>0.86</td>
</tr>
<tr>
<td>25</td>
<td>.58</td>
<td>.58</td>
<td>142.3</td>
<td>2.3</td>
</tr>
<tr>
<td>30</td>
<td>.52</td>
<td>.52</td>
<td>159.2</td>
<td>2.1</td>
</tr>
<tr>
<td>35</td>
<td>.55</td>
<td>.55</td>
<td>178.61</td>
<td>2.26</td>
</tr>
<tr>
<td>40</td>
<td>.54</td>
<td>.54</td>
<td>208.4</td>
<td>2.25</td>
</tr>
<tr>
<td>45</td>
<td>.38</td>
<td>.38</td>
<td>235.58</td>
<td>1.46</td>
</tr>
<tr>
<td>50</td>
<td>.52</td>
<td>.52</td>
<td>309.28</td>
<td>2.86</td>
</tr>
</tbody>
</table>

Note: $B_{cost}$ and $S_{cost}$ denote the reduction in expected cost using the brute-force and proposed sub-optimal approach respectively.

5.2. Comparison against other sub-optimal approaches

Section 5.1 showed the effectiveness of the proposed approach compared to exhaustive search in terms of complexity reduction and performance. In this subsection, we compare the proposed approach to other sub-optimal approaches of generating control policies with best expected performance for a family of Markov chains. We consider the sub-optimal approach presented in $^7$ where optimal policies for each individual Markov chain in the family are computed based on dynamic programming principles and the performance of each generated policy over the family of Markov chains is compared to select the best policy. We will term the approach of $^7$ as the $L_{min}$ approach. The Dynamic Programing approach for average cost infinite horizon policy generation based on policy iteration will involve an $N \times N$ matrix inversion at each step and thus has a stepwise computational complexity of $O(N^3)$. If the family of Markov chains consists of $L$ Markov chains, then $L$ optimal policies will be generated and each policy will be evaluated $L$ times, consisting of a total of $L^2$ evaluations. The evaluation for each policy will have complexity of $O(N^3)$ based on calculations similar to policy iteration. Thus, if the policy iteration converges in few steps, the complexity of this approach will be $O(L^2N^3)$. However, if the number
of steps required in its convergence is of the order of $N$, then the complexity will be $O(LN^4 + L^2N^3)$. Thus, this approach has comparable complexity to our presented approach when $L < N$, and higher complexity for $L > N$. Furthermore, by limiting its search to $L$ nominal policies, the $L_{min}$ approach can frequently miss out on the best expected performance policy that is captured by the proposed approach. We next present an example to illustrate this behavior.

Let us consider an example with $n = 1$ and thus $N = 2^n = 2$. Let the family of Markov chains without control be

\[
M^u_1 = \begin{pmatrix} 0.9000 & 0.1000 \\ 0.7000 & 0.3000 \end{pmatrix}
\]

\[
M^u_2 = \begin{pmatrix} 0.0900 & 0.9100 \\ 0.1117 & 0.8883 \end{pmatrix}
\]

and let the corresponding Markov chains with control be

\[
M^c_1 = \begin{pmatrix} 0.6000 & 0.4000 \\ 0.5000 & 0.5000 \end{pmatrix}
\]

\[
M^c_2 = \begin{pmatrix} 0.1363 & 0.8637 \\ 0.6787 & 0.3213 \end{pmatrix}
\]

Each Markov chain has states 0 and 1 and the desired control outcome is to minimize the steady-state probability of state 0. There are a total of $2^2 = 4$ possible control policies. The steady state probabilities of state 0 of the system \(\{M^u_1, M^c_1\}\) under all possible control policies are shown in Table 2. The system \(\{M^u_1, M^c_1\}\) denotes a network whose behavior under no-control can be represented by the single Markov chain $M^u_1$ and the behavior under control can be represented by $M^c_1$.

<table>
<thead>
<tr>
<th>ControlPolicy</th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSP</td>
<td>0.8750</td>
<td>0.8333</td>
<td>0.6364</td>
<td>0.5556</td>
</tr>
</tbody>
</table>

Similarly, the steady-state probabilities of state 0 for the system \(\{M^u_2, M^c_2\}\) is shown in Table 3.

<table>
<thead>
<tr>
<th>ControlPolicy</th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSP</td>
<td>0.1093</td>
<td>0.4272</td>
<td>0.1145</td>
<td>0.4400</td>
</tr>
</tbody>
</table>
Since the control goal is to minimize the steady-state probability of state 0, the average cost over the family of Markov chains is the average steady state probability of state 0. The average cost for the entire family under the different control policies is shown in Table 4.

<table>
<thead>
<tr>
<th>ControlPolicy</th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>AvgSSP</td>
<td>0.4922</td>
<td>0.6303</td>
<td>0.3745</td>
<td>0.4978</td>
</tr>
</tbody>
</table>

From Table 2, the optimal nominal policy for the system \( \{M_1, M_1\} \) is 11 as it minimizes the probability of state 0. Similarly from Table 3, the optimal nominal policy for the system \( \{M_2, M_2\} \) is 00 as it minimizes the probability of state 0. The \( L_{min} \) approach, which essentially selects from the individual optimal policies, will select policy 00 as it has the lowest cost among the policies 00 and 11 as shown in Table 4. In this example, the policy generated by the \( L_{min} \) method \(^7\) does not produce the optimal average policy 10. However, the proposed method will provide the optimal policy: starting from policy 00, the proposed algorithm would find that adding control in state 0, thus producing policy 10, would be the most advantageous step, at which point no further improvements could be found. Thus, the proposed algorithm is capable of finding policies stronger than the \( L_{min} \) method in many scenarios, which in turn motivates its development.

5.3. Biologically Motivated Example

In recent literature, a number of studies have been published on intra-tumor heterogeneity, which denotes the presence of multiple clones on a single tumor mass, and inter-tumor heterogeneity, which denotes dissimilar pathway activations in different metastatic sites of a single patient \(^{18,19,20}\). Thus, the success of personalized cancer therapy on an intra-tumor heterogeneous mass will depend on targeting the genes and proteins that are used by tumors to sustain and develop heterogeneity \(^{21}\). The composite model of a heterogeneous tumor will consist of a combination of individual models of tumorous genetic regulatory networks. This can be considered equivalent to the case of controlling a collection of Markov chains. We next consider a biological scenario that shows the effectiveness of our new algorithm compared to the \( L_{min} \) approach.

A gene or protein can behave as an oncogene or a tumor suppressor based on the biological context \(^{22}\). For instance, \( E\text{-cadherin} \) has been known to behave as a tumor suppressor \(^{23}\) as well as correlate with tumor progression \(^{24}\). Thus a heterogeneous tumor can contain a set of cells where a protein \( P_X \) acts as an oncogene and another set of cells where the protein \( P_X \) acts as a tumor suppressor. If
we apply a targeted drug to inhibit $P_X$, it will have different phenotypic changes on the two sets of cells. Consider a synthetic biological system consisting of a collection of cells where the GRN is modeled by a $n = 7$ gene Markov chain with $L = 200$. The control action is inhibition of a protein via drug application. The first gene indicating the most significant bit (MSB) of the binary representation of the genetic profile is considered to be representative of the cancer phenotype. Thus, states 1 (corresponding to 0000000) to 64 (corresponding to 0111111) are the desirable states and states 65 (corresponding to 1000000) to 128 (corresponding to 1111111) are undesirable states. The tumor consists of the following two cell types:

(Type A) The control protein acts as a tumor suppressor in this category of cells and thus application of control protein moves the cells towards progression of cancer. Without control action, the tumor suppressor is functional and cells are primarily in the desirable states. Sample transition probability matrices of the Markov chains representing the cell without control action and with control actions are shown in figures 1 and 2 respectively. The Markov chains are shown as images with heat maps denoting the transition probabilities.

(Type B) The control protein acts as an oncogene in this category of cells and thus application of control protein inhibits the progression of cancer. Sample transition probability matrices of the Markov chains to represent the cell without control action and with control actions are shown in figures 3 and 4 respectively.

We randomly generated 200 transition probability matrices for type A and type B with and without control action. For generating the random transition probability matrices of type A and type B, we considered another parameter $\chi_1$ which denotes the average ratio of the probability of transition to an undesirable state for
uncontrolled (controlled) matrices of Type A (Type B). Similarly, the parameter $\chi_2$ denotes the average ratio of the probability of transition to a desirable state for uncontrolled (controlled) matrices of Type B (Type A). For instance, in Fig 1, $\chi_1 = 0.1$ which denotes that $E(\sum_{j=65}^{128} P(i,j))/E(\sum_{j=1}^{64} P(i,j)) = 0.1$ where the expectation is taken over $i = \{1, \cdots, 128\}$. We applied our proposed approach and the $L_{\min}$ approach to maximize the steady state probabilities of states 1 to 64. For $n = 7$, $L = 200$ and $\chi_1 = \chi_2 = 0.1$, our proposed approach was able to produce a 0.6067 steady state probability for MSB=0 whereas the $L_{\min}$ approach produced
a 0.5 steady state probability for MSB=0. The time taken by the proposed approach was 98.9 seconds whereas \( L_{\text{min}} \) approach required 188.05 seconds. The calculations were completed using MATLAB R2011b on a Windows 64-bit machine with Intel Core i5 CPU @ 2.53 GHz and 4GB RAM. The MATLAB MDP toolbox was used to implement the dynamic programming algorithm for the \( L_{\text{min}} \) approach. The performance results for \( L_{\text{min}} \) and the proposed approach \( PA \) for different numbers of genes \((n)\) and different sizes of families \((L)\) are shown in Table 5.

Table 5. Performance comparison for \( L_{\text{min}} \) and proposed algorithm \((PA)\) for different \( n, L, \chi_1 \) and \( \chi_2 \).

<table>
<thead>
<tr>
<th>( n )</th>
<th>( L )</th>
<th>( \chi_1 )</th>
<th>( \chi_2 )</th>
<th>( PA_{ss} ) (s)</th>
<th>( PA_{time} ) (s)</th>
<th>( L_{\text{min}}_{ss} )</th>
<th>( L_{\text{min}}_{time} ) (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>100</td>
<td>0.03</td>
<td>0.15</td>
<td>0.696</td>
<td>1.76</td>
<td>0.551</td>
<td>2.41</td>
</tr>
<tr>
<td>5</td>
<td>100</td>
<td>0.01</td>
<td>0.1</td>
<td>0.723</td>
<td>1.9</td>
<td>0.541</td>
<td>2.5</td>
</tr>
<tr>
<td>5</td>
<td>100</td>
<td>0.1</td>
<td>0.1</td>
<td>0.606</td>
<td>1.44</td>
<td>0.50</td>
<td>2.89</td>
</tr>
<tr>
<td>5</td>
<td>100</td>
<td>0.05</td>
<td>0.1</td>
<td>0.656</td>
<td>1.69</td>
<td>0.522</td>
<td>3.30</td>
</tr>
<tr>
<td>6</td>
<td>100</td>
<td>0.1</td>
<td>0.1</td>
<td>0.607</td>
<td>9.1</td>
<td>0.51</td>
<td>10.2</td>
</tr>
<tr>
<td>6</td>
<td>150</td>
<td>0.05</td>
<td>0.05</td>
<td>0.644</td>
<td>13.42</td>
<td>0.50</td>
<td>17.81</td>
</tr>
<tr>
<td>6</td>
<td>200</td>
<td>0.01</td>
<td>0.05</td>
<td>0.711</td>
<td>19.03</td>
<td>0.519</td>
<td>31.03</td>
</tr>
<tr>
<td>7</td>
<td>200</td>
<td>0.1</td>
<td>0.1</td>
<td>0.6067</td>
<td>98.9</td>
<td>0.5</td>
<td>188.05</td>
</tr>
</tbody>
</table>

Note: \( PA_{ss} \) and \( L_{\text{min}}_{ss} \) denotes the steady state probabilities of desirable states (states 1 to 64) following the application of the proposed policy and \( L_{\text{min}} \) policy respectively.

Fig. 4. Transition Probability Matrix for type B with control
6. Conclusions

In this paper, we presented a new algorithm for the calculation of stationary control policies with best expected performance for a family of Markov chains. For families of Markov chains, especially larger ones, the proposed algorithm is capable of generating sub-optimal average case stationary policies in a computationally efficient manner by sequentially adding states that produces the maximum reduction in cost. Due to its sub-optimal nature, this algorithm cannot guarantee the optimal solution but as our simulation results show, it does produce results close to optimal in a variety of situations. Through simulations and examples, we showed that this approach can produce better results in many scenarios as compared to the sub-optimal approach of selecting the best policy from individual optimal policies. Thus, the proposed approach can be a valuable tool along with other approaches for generation of stationary control policies with best expected performance for a family of Markov chains.

Acknowledgements

This research was supported by the National Science Foundation under Grant CCF 0953366.

References