Approximating Large-Scale Binary Integer Programs by Discrete Optimal Control

Pengbo Zhang

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

University of Washington

2015

Reading Committee:

Zelda B. Zabinsky, Chair

Wolf Kohn, Chair

Archis Ghate

Program Authorized to Offer Degree:
Industrial & Systems Engineering

University of Washington

Abstract

Approximating Large-Scale Binary Integer Programs by Discrete Optimal Control

Pengbo Zhang

Co-Chairs of the Supervisory Committee:
Professor Zelda B. Zabinsky
Department of Industrial & Systems Engineering

Research Professor Wolf Kohn
Department of Industrial & Systems Engineering

Optimal control theory has been introduced as a powerful tool for approximately solving binary integer programming problems. In previous studies, an approach using continuous optimal control theory was developed, where the original binary integer problem was transformed into a continuous linear quadratic problem. However, the continualization process added approximation error.

The primary objective of this dissertation is to develop a discrete optimal control technique to approximately solve large-scale binary integer programming problems efficiently. The basic idea of the new algorithm is to transform the original binary integer problem into the form of a discrete linear quadratic tracking problem to take advantage of control theory properties and build good numerical stability properties. Because the time index in the reformulation of the binary integer problem represents the dimension of the problem, a discrete time approach more accurately represents the partial summing reformulation than the continuous approach. Also, instead of iteratively solving the linear quadratic tracking problem based on a reduced set of constraints as done previously, the linear quadratic tracking problem is solved only once for the full-state space. The advantage of solving the linear quadratic tracking problem once is that no error is introduced by the constraint reduction. However, if the number of constraints is very large, the full-state space may require too

much computation, so a stochastic decomposition method, which is an extension of the constraint reduction technique, is also developed.

Binary integer problem is NP-hard, and even finding a feasible solution is considered NP-complete. In this dissertation, an idea of balancing the feasibility problem with optimizing the objective function has been explored to improve the approximate solution. The approach taken is to apply the feasibility pump idea in the form of discrete optimal control toward finding a feasible solution.

Several other variations of the full-state space discrete control problem, including combinations of cross-entropy method, bang-bang control method, minimum fuel formulation, and minimum-variance formulation, are studied and presented to reveal their potential for approximately solving binary integer problems.

The computational results show the effectiveness of the proposed approaches for approximating solutions to large-scale binary integer programs.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisors, Dr. Zelda Zabinsky and Dr. Wolf Kohn, who have supported me professionally and personally throughout the graduate school years and especially in completing this thesis work. I would also like to thank my committee members, Dr. Archis Ghate and Dr. Rekha Thomas, for their feedback and valuable suggestions which improved my work.

I am very thankful to the wonderful professors in the Department of Industrial and Systems Engineering that have provided me knowledge, skills and attitude in my research and everyday life: Prof. Zelda Zabinsky, Prof. Wolf Kohn, Prof. Archis Ghate, Prof. Kailash Kapur, Prof. Richard Storch, Prof. Christina Mastrangelo, Prof. Benita Beamon, Prof. Linda Boyle, Prof. Art Chaovalitwongse, Prof. Joe Heim and Prof. Vladimir Brayman. I am also thankful to the ISE staff who have offered me dedicated help through these years: Jennifer Tsai, Deborah Fromm, Kellus Stone and Sheila Prusa.

I would like to thank all my friends at the ISE department, at the UW and in Seattle, whose friendship made the gloomy days in Seattle shining and enjoyable. Special thanks go to Yanfang Shen, Hongrui Liu and Wei Wang, for their help, advice and encouragement.

I cannot thank enough to my dearest parents, my brother and my parents-in-law, for their endless love and continuous support in my life.

Finally, many thanks and love go to my husband, Zheng Liu, for always being my best friend and supporter, and our two sons, Ryan and Alexander, who bring me a lot of joy and happiness.

This research has been supported in part by the National Science Foundation through Grant CMMI-0908317, and by Microsoft Dynamics, Business Solutions.

DEDICATION

To my husband and our two boys

Chapter 1

INTRODUCTION

1.1 Motivation

Many decision-making problems involving "yes" or "no" decisions can be formulated as binary integer programming (BIP) problems, i.e., a set of linear constraints together with a linear objective function and a restriction of the variables to 0 or 1. Common solution techniques for BIP problems are based on solving its linear relaxation together with some branching strategy, such as branch-and-bound and cutting-plane method [43]. These methods can work very well for small problems and also for larger problems with special structure, however, they are computationally expensive to solve general large-scale BIP problems.

In a previous study [41, 42], an approach to approximating a BIP solution using continuous optimal control theory was introduced and showed promise for large-scale problems. The key innovation to the optimal control approach is to map the vector of n binary decision variables into a scalar function defined over a time interval $[0, n]$ and define a linear quadratic tracking (LQT) problem that can be solved efficiently. The research presented in this dissertation uses the same mapping, but instead of solving the LQT problem in continuous time, this dissertation explores solving the LQT problem in discrete time, because the time index in the reformulation of the BIP represents the dimension of the problem, $\{0, 1, \ldots, n\}$, and a discrete time approach more accurately represents the partial summing formulation than the continuous approach.

1.2 Contribution

The research presented in this dissertation makes a contribution by developing discrete optimal control algorithms with low-polynomial time complexity for approximating solutions to large-scale BIP problems.

Transforming the original BIP problem into the form of a discrete time LQT formulation

provides advantages over the continuous time LQT formulation not only in reducing computation but also in generating a better approximate solution. In Chapter 3, the discrete LQT formulation is proven to have an optimal binary solution, analogous to a classical bang-bang control in continuous time. It suggests an algorithmic approach for solving the discrete LQT problem.

If the number of constraints is too large, then the full-state space may require too much computation. So, in Chapter 4, a stochastic decomposition method is developed to help reduce the computational complexity. Motivated by Aoki's partitioning method [3], the proposed algorithm decouples the problem through the state variables so that the large-scale problem can be transformed into smaller subproblems which are easier to solve. And stochastic terms are introduced to account for the error due to state dimensionality reduction and problem relaxation. Then, the solutions of the smaller stochastic subproblems are used to obtain a good approximate solution for the original problem.

The experience with decomposing the LQT problem for the BIP can be generalized to any LQT problem with large state space. The generalized stochastic decomposition method is also presented and discussed in Chapter 4.

The discrete LQT formulation in Chapter 3 provides approximate solutions that may be infeasible and/or "superoptimal". Motivated by infeasible-path interior point methods for linear programming, also known as primal-dual interior point methods [44] that have been used to balance feasibility and optimality, in Chapter 5, two coupled linear quadratic Gaussian tracking (LQGT) problems are defined: one named optimality LQGT problem that tracks all the constraints as well as the objective function, and the other named feasibility LQGT problem that only tracks the constraints. The two LQGT problems are iteratively solved in sequence to improve the feasibility and optimality measures with respect to the original BIP problem. By applying the feasibility pump idea [17] in the form of discrete optimal control to this BIP, the approximate solution regarding the feasibility metric has been further improved.

The bang-bang control derived in Chapter 3 has two computational difficulties, one is the presence of singular arcs where the bang-bang control is not specified, and the second is the need to solve a two-point boundary-value problem, since the boundary conditions

required for solution are the initial state and the final costate. Therefore, several other approaches including the minimum fuel formulation, minimum-variance formulation, and the cross-entropy method, are considered to be combined with the bang-bang control method to approximately solve the BIP. These approaches are studied and presented in Chapter 6. Even though the computational performances of these algorithms are not as good as the algorithms introduced in other chapters, they are still relatively effective and provide experiences that may lead to further improvements.

1.3 Organization

The rest of this dissertation is organized as follows. Chapter 2 gives a literature review of some BIP algorithms and decomposition algorithms; Chapter 3 develops a discrete LQT formulation method to approximating the BIP solution; Chapter 4 presents a stochastic decomposition method to help reduce the computational complexity for approximating solutions to BIP problems, and then generalizes the method to any LQT problem with large state space. Chapter 5 develops a stochastic methodology to balance the feasibility problem with optimizing the objective function, and further improves its performance by applying a feasibility pump idea; Chapter 6 studies several other approximation algorithms and their performances for solving large-scale BIP problems. A summary is given in Chapter 7 and future work is also discussed.

Chapter 2

BACKGROUND AND LITERATURE REVIEW

2.1 Algorithms for BIP Problems

Many decision problems in economics and engineering can be formulated as BIP problems. These BIP problems are often easy to state but difficult to solve due to the fact that many of them are NP-hard [43], and even finding a feasible solution is considered NP-complete [13] [22]. Because of their importance in formulating many practical problems, BIP algorithms have been widely studied. These algorithms can be classified into exact and approximate algorithms as follows [7]:

(1) Exact algorithms: The exact algorithms are guaranteed either to find an optimal solution or prove that the problem is infeasible, but they are usually computationally expensive. Major methods for BIP problems include branch and bound [33], branch-and-cut [11], branch-and-price [5], dynamic programming methods [26], and semidefinite relaxations [23].

(2) Approximate algorithms: The approximate algorithms are used to achieve efficient running time with a sacrifice in the quality of the solution found. Examples of well-known metaheuristics, as an approximate approach, are simulated annealing [24], annealing adaptive search [45], cross entropy [37], genetic algorithms [20] and nested partitions [39]. Moreover, many hybrid approaches that combine both the exact and approximate algorithms have been studied to exploit the benefits of each [21]. For additional references regarding large-scale BIP algorithms, see [19, 30, 38, 43].

Today, branch-and-cut is the state-of-the-art method to solve BIP problems and has been successfully implemented by the vast majority of solvers [4, 28, 32].

2.2 Algorithms for BIP Problems by Optimal Control

Effective heuristic techniques that transform binary optimization problems into problems falling in the control theory domain have been studied recently. The first exploration of control theory to BIP was done by Littleton [29] in 2005. Littleton's approach used continualization idea to develop an equivalent problem and then used a Taylor's series expansion to approximate the differential equations. In [41, 42], an approach to approximating a BIP solution using continuous optimal control theory was developed, where the BIP was transformed into a continuous LQT problem. There were limited success with these approaches because the continualization process added approximation error. However, the idea to solve a control problem to approximate the BIP problem shows promise for large-scale problems.

Attempts at transforming optimization problems into problems falling in the control theory or signal processing domain have also recently been tried elsewhere (e.g., [9, 10]). In some cases the transformation may, in principle, let the optimization problem be solved in hardware, which can dramatically decrease the computation time.

This dissertation explores the idea to approximate a BIP solution using discrete LQT-based methods procedure to take advantage of control theory properties and build good numerical stability properties.

2.3 Algorithms for Decomposition Methods

Decomposition methods have been developed for linear programming and BIP, for problems with large number of variables and constraints. Traditional decomposition methods, such as Dantzig-Wolfe method [14] and the Lagrangian method [16] as applied to an integer program, aim at providing a tight linear programming relaxation bound. Decomposition for optimal control, as described by Aoki [3] has also been developed.

In this dissertation, a stochastic decomposition method based on constraint reduction technique is presented to reduce the computational complexity to approximating a BIP solution. It is then generalized to any LQT problem with large state space.

2.4 Feasibility Problem for BIP Problems

It is known that finding a feasible solution for a generic mixed integer program (MIP) problem is NP-hard. In [17], Fischetti, Glover, and Lodi developed a feasibility pump heuristic for quickly finding high quality feasible solutions of MIP problems. The feasibility pump approach starts from the relaxed linear program optimum and computes two trajectories of points that satisfy MIP feasibility in a complementary but partial way — one satisfies the integer requirement and the other the linear constraints. The distance between pairs of points on the two trajectories decreases monotonically until the method cycles, unable to further decrease the distance. If the distance is reduced to zero before cycling, then a feasible point has been obtained. Otherwise a random perturbation is used to restart the method from a new point.

This heuristic turned out to be very successful in finding feasible solutions even for very hard MIP instances. By applying the feasibility pump idea in the form of discrete optimal control, a new algorithm is developed in this dissertation to balance the feasibility problem with optimizing the objective function. It iteratively improves the feasibility metric and the optimality metric.

Chapter 3

A DISCRETE LQT-BASED APPROXIMATION PROCEDURE FOR SOLVING LARGE-SCALE BIP PROBLEMS

In this chapter, a discrete LQT formulation is developed to approximately solve large-scale BIP problems efficiently. The key idea is to map the vector of n binary decision variables into a scalar function defined over a time interval $[0, n]$ and construct an LQT problem that can be solved efficiently. In Theorem 1, it is proved that an LQT formulation has an optimal binary solution, analogous to a classical bang-bang control in continuous time. The LQT approach can provide advantages in reducing computation while generating a good approximate solution. Numerical examples are presented to demonstrate the usefulness of the proposed method. Most of Chapter 3 has been published in [46].

3.1 Discrete LQT Formulation

The original BIP problem considered here is:

$$\min_{\substack{u_j \\ j=0,\ldots,n-1}} \quad \sum_{j=0}^{n-1} \tilde{c}_j u_j \tag{3.1}$$

$$\text{s.t.} \quad \sum_{j=0}^{n-1} \tilde{a}_{ij} u_j = \tilde{b}_i \quad i = 1, \ldots, m \tag{3.2}$$

$$u_j \in \{0, 1\} \qquad j = 0, \ldots, n-1 \tag{3.3}$$

where u_j for $j = 0, \ldots, n-1$ are binary decision variables. It is assumed that $\tilde{c}_j, \tilde{a}_{ij}$, and $\tilde{b}_i$ are real known values for $i = 1, \ldots, m$ and $j = 0, \ldots, n-1$ and there exists at least one feasible point.

3.1.1 Partial Summing Formulation

Define partial summing variables as in [41] from the original BIP problem as

$$f_{0,j+1} = f_{0,j} + \tilde{c}_j u_j \tag{3.4}$$

$$f_{i,j+1} = f_{i,j} + \tilde{a}_{ij} u_j \tag{3.5}$$

for $i = 1, \ldots, m$ and $j = 0, \ldots, n-1$, with initial conditions $f_{0,0} = f_{i,0} = 0$.

For ease of notation, create a new $(m+1) \times 1$ vector x_j as

$$x_j = \begin{bmatrix} f_{0,j} \\ f_{1,j} \\ \vdots \\ f_{m,j} \end{bmatrix}$$

and the i^{th} element of x_j is denoted $x_{j(i)}$ for $i = 1, \ldots, m+1$ and for $j = 0, \ldots, n$. Define the $(m+1) \times 1$ vector a_j for $j = 0, \ldots, n-1$, and the $(m+1) \times 1$ vector b as

$$a_j = \begin{bmatrix} \tilde{c}_j \\ \tilde{a}_{1j} \\ \vdots \\ \tilde{a}_{mj} \end{bmatrix}, \; b = \begin{bmatrix} 0 \\ \tilde{b}_1 \\ \vdots \\ \tilde{b}_m \end{bmatrix} \tag{3.6}$$

where the i^{th} element of b is denoted as $b_{(i)}$ for $i = 1, \ldots, m+1$. The partial summing formulation, with initial conditions x_0 as a vector of zeros, is developed as follows:

$$\begin{aligned}
\min_{\substack{u_j \\ j=0,\ldots,n-1}} \quad & x_{n(1)} \\
\text{s.t.} \quad & x_{j+1} = x_j + a_j u_j \qquad j = 0, \ldots, n-1 && (3.7) \\
& x_0 = 0 && (3.8) \\
& x_{n(i)} = b_{(i)} \qquad i = 2, \ldots, m+1 && (3.9) \\
& u_j(u_j - 1) = 0 \qquad j = 0, \ldots, n-1. && (3.10)
\end{aligned}$$

Proposition 1 *The partial summing formulation (3.7)-(3.10) is equivalent to the original BIP problem (3.1)-(3.3).*

The proof is straight-forward; the constraints ensure feasibility and the objective function is equivalent to the original BIP problem (3.1)-(3.3).

3.1.2 Construction of the LQT Problem

Construct a discrete LQT problem by first defining an error term, as a measure of unsatisfied constraints, an $(m+1) \times 1$ vector e_j for $j = 0, \ldots, n$, as

$$e_j = x_j - b. \tag{3.11}$$

Develop the dynamics in terms of the measure e_j, by combining (3.11) with (3.7), yielding

$$e_{j+1} = e_j + a_j u_j \tag{3.12}$$

and note that $e_0 = -b$, given initial conditions $x_0 = 0$. The criterion is to minimize the measure of unsatisfied constraints using a terminal penalty for feasibility and objective function value, which is given by

$$J(u) = \frac{1}{2} \sum_{j=0}^{n-1} e_j^T Q_j e_j + \frac{1}{2} e_n^T F e_n. \tag{3.13}$$

The parameters Q_j and F are positive semi-definite matrices and user-specified. The $(m+1) \times (m+1)$ matrix Q_j is used to penalize the unsatisfied constraints. The $(m+1) \times (m+1)$ matrix F is used to penalize the terminating conditions and aide in minimizing the original objective function.

The discrete LQT problem with the criterion in (3.13), and relaxing (3.10), is as follows:

$$\min_{\substack{u_j \\ j=0,\ldots,n-1}} \quad \frac{1}{2} \sum_{j=0}^{n-1} e_j^T Q_j e_j + \frac{1}{2} e_n^T F e_n \tag{3.14}$$

$$\text{s.t.} \quad e_{j+1} = e_j + a_j u_j \quad j = 0, \ldots, n-1 \tag{3.15}$$

$$0 \leq u_j \leq 1 \qquad j = 0, \ldots, n-1 \tag{3.16}$$

$$e_0 = -b. \tag{3.17}$$

3.1.3 Existence of Binary Solution to LQT Problem

Whereas the partial summing formulation (3.7)-(3.10) is equivalent to the original BIP problem (3.1-3.3), the discrete LQT problem (3.14)-(3.17) approximates the partial sum-

ming formulation (3.7)-(3.10) in two ways. First, the feasibility condition with $x_{n(i)} = b_{(i)}$, as in constraint (3.9), is put into the criteria with penalty parameters F and Q_j. Second, the binary constraint $u_j \in \{0, 1\}$, is relaxed as $0 \leq u_j \leq 1$, as in constraint (3.16).

It is known that solving the discrete LQT problem (3.14)-(3.17) directly is numerically unstable [27]. However, Theorem 1 suggests an algorithmic approach to solve the discrete LQT problem (3.14)-(3.17), by making a discrete analog to a bang-bang control with a switching function.

Theorem 1 *Analogous to a bang-bang control in continuous time, the discrete LQT problem (3.14)-(3.17) has an optimal binary solution with* $u_j \in \{0, 1\}$ *for discrete times* $j = 0, 1, \ldots, n-1$ *with non-singular arcs.*

Proof. First, construct the Hamiltonian function [27] as follows

$$H(e_j, \lambda_{j+1}, u_j) = \frac{1}{2} e_j^T Q_j e_j + \lambda_{j+1}^T (e_j + a_j u_j) \tag{3.18}$$

where λ_j is the $(m+1) \times 1$ costate vector, for $j = 0, \ldots, n-1$, and it satisfies

$$\lambda_j = \lambda_{j+1} + Q_j e_j \text{ and } \lambda_n = F e_n. \tag{3.19}$$

Let e^*, λ^* and u^* be the optimal solution, by the necessary conditions for optimality [27],

$$\begin{aligned} & H(e_j^*, \lambda_{j+1}^*, u_j^*) \leq H(e_j^*, \lambda_{j+1}^*, u_j) \\ & \Rightarrow \frac{1}{2} e_j^{*T} Q_j e_j^* + \lambda_{j+1}^{*T} \left(e_j^* + a_j u_j^*\right) \leq \frac{1}{2} e_j^{*T} Q_j e_j^* + \lambda_{j+1}^{*T} \left(e_j^* + a_j u_j\right) \\ & \Rightarrow \lambda_{j+1}^{*T} a_j u_j^* \leq \lambda_{j+1}^{*T} a_j u_j, \quad \forall u_j \in [0, 1]. \end{aligned} \tag{3.20}$$

Thus,

$$u_j^* = \begin{cases} 1 & \text{if } \lambda_{j+1}^{*T} a_j < 0, \\ \in [0, 1] & \text{if } \lambda_{j+1}^{*T} a_j = 0, \\ 0 & \text{if } \lambda_{j+1}^{*T} a_j > 0. \end{cases} \tag{3.21}$$

■

If $\lambda_{j+1}^{*T} a_j \neq 0$, binary values for u_j^* are determined by (3.21). When $\lambda_{j+1}^{*T} a_j = 0$, the arc is singular, and reintroduce constraint (3.10), $u_j(1 - u_j) = 0$, to force a binary solution.

To get an intuitive understanding of the singularity issue, suppose all $Q_j = 0$, and the element at row 1, column 1 of matrix F equals zero. Then the discrete LQT problem (3.14)-(3.17) reduces to minimize the infeasibility penalty term, $\frac{1}{2}\sum_{i=1}^{m}\left[\left(\sum_{k=0}^{n-1}\tilde{a}_{ik}u_k - \tilde{b}_i\right)^2 F_i\right]$. If this term equals zero, then $e_n = 0$, satisfying all of the original constraints (3.3), and $\lambda_n = 0$ from (3.19), and because $Q_j = 0$, all $\lambda_j = 0$. Then $\lambda_{j+1}^{*T} a_j = 0$ for all j. However, if Q_j and the first element of F have positive values, then $\lambda_{j+1}^{*T} a_j$ may be positive or negative and (3.21) is useful.

3.2 Solutions to the LQT Problem

To create an LQT problem that is practical to solve, a penalty term $u_j(u_j - 1)R_j$ is introduced in the criterion, where R_j is a Lagrangian multiplier associated with constraint (3.10). The full-state space LQT problem is:

$$\min_{\substack{u_j \\ j=0,\ldots,n-1}} \quad \frac{1}{2}\sum_{j=0}^{n-1}\left(e_j^T Q_j e_j + u_j(u_j-1)R_j\right) + \frac{1}{2}e_n^T F e_n \tag{3.22}$$

$$\text{s.t.} \quad e_{j+1} = e_j + a_j u_j \qquad j = 0,\ldots,n-1 \tag{3.23}$$

$$e_0 = -b. \tag{3.24}$$

Here, the criterion in (3.22) is to minimize the measure of unsatisfied constraints using a terminal penalty for infeasibility and objective function value, and R_j is a positive scalar. The parameters a_j, Q_j and F have the same forms as a_j, Q_j and F in the discrete LQT problem (3.14)-(3.17).

There are two ways to solve the full-state space LQT problem (3.22)-(3.24), one is the dynamic programming method, the other is the Lagrangian method. The two methods are discussed in the following subsections in detail.

3.2.1 Dynamic Programming Method

The optimal control for the full-state space LQT problem (3.22)-(3.24) $\hat{u}_j$ can be solved by the standard dynamic programming method [6] (see appendix for details). The computation associated with solving the full-state space LQT problem (3.22)-(3.24) is $O(nm^3)$. The

approximate binary solution to the original BIP problem (3.1)-(3.3) is as follows:

$$u_j^* = \begin{cases} 0 \text{ for } \hat{u}_j < 0.5 \\ 1 \text{ for } \hat{u}_j \geq 0.5 \end{cases} \tag{3.25}$$

for $j = 0, 1, \ldots, n-1$.

Motivated by the successive overrelaxation method [40], a weighting factor ω is introduced to improve the stability of the proposed method. Rather than applying quantization at the final step as shown in equation (3.25), apply quantization at each step and propagate the binary value $\bar{u}_j$ during the dynamic programming procedure (see Appendix A for details). At the final step, replace $\hat{u}_j$ in equation (3.25) with

$$\omega\hat{u}_j + (1-\omega)\bar{u}_j \tag{3.26}$$

to get the approximate binary solution.

The flowchart of the discrete LQT-based approximation procedure with dynamic programming for solving large-scale BIP problems is shown in Figure 3.1.

3.2.2 The Lagrangian Method

An alternative for solving the full-state space LQT problem (3.22)-(3.24) with a dynamic programming approach is to use the Lagrangian method to find the optimal u_j. The Lagrangian method introduces Lagrange multipliers and solves necessary optimality conditions directly.

The Lagrange multipliers associated with the full-state space LQT problem (3.22)-(3.24) are denoted by $\lambda_1, \cdots, \lambda_n$ and the Lagrangian function $\Lambda(e_0, \cdots, e_n, u_0, \cdots, u_{n-1}, \lambda_1, \cdots, \lambda_n)$ is given by

$$\begin{aligned} \Lambda(e_0, \cdots, e_n, u_0, \cdots, u_{n-1}, \lambda_1, \cdots, \lambda_n) =& \frac{1}{2}\sum_{j=0}^{n-1}\left[e_j^T Q_j e_j + (u_j^2 - u_j)R_j\right] + \frac{1}{2}e_n^T F e_n \\ &+ \sum_{j=0}^{n-1}\lambda_{j+1}^T\left(-e_{j+1} + e_j + a_j u_j\right). \end{aligned}$$

And therefore a necessary condition of optimality is that $d\Lambda = 0$. Now, since the Lagrangian Λ is a function of three sets of optimal variables e_j, u_j, and λ_{j+1}, take the partial derivatives

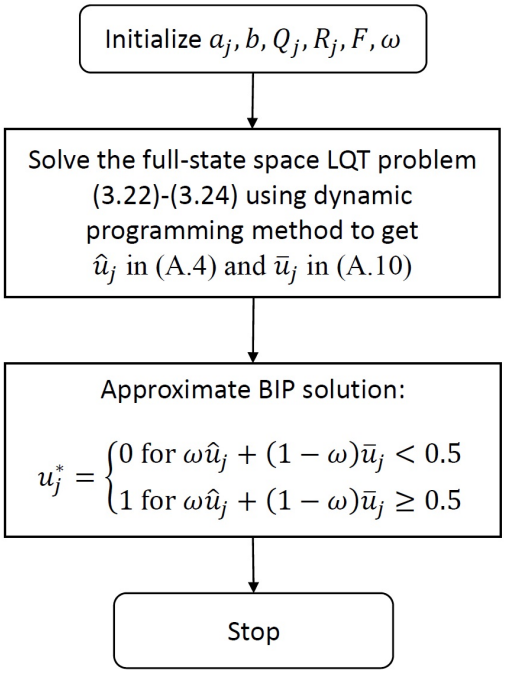

Figure 3.1: Flowchart of the discrete LQT-based procedure for approximating BIP solutions

of Λ with respect to each of the variables e_j, u_j, and λ_{j+1} for $k = 0, \ldots, n-1$ and set them to zero, which yields the following equations:

$$\begin{cases} 0 = \frac{\partial \Lambda}{\partial e_j} = Q_j e_j + \lambda_{j+1} - \lambda_j & j = 0, \cdots, n-1 \\ 0 = \frac{\partial \Lambda}{\partial e_n} = F e_n - \lambda_n \\ 0 = \frac{\partial \Lambda}{\partial u_j} = (u_j^T - \frac{1}{2}) R_j + \lambda_{j+1}^T a_j & j = 0, \cdots, n-1 \\ 0 = \frac{\partial \Lambda}{\partial \lambda_{j+1}} = -e_{j+1} + e_j + a_j u_j & j = 0, \cdots, n-1. \end{cases} \tag{3.27}$$

Solving above equations in (3.27), yields

$$\lambda_j = \lambda_{j+1} + Q_j e_j \tag{3.28}$$

$$\lambda_n = F e_n \tag{3.29}$$

$$u_j = \frac{1}{2} - R_j^{-1} a_j^T \lambda_{j+1} \tag{3.30}$$

$$e_{j+1} = e_j + a_j u_j. \tag{3.31}$$

These equations provide a solution to the full-state space LQT problem (3.22)-(3.24),

however, it would be more convenient if u_j has a linear expression in terms of e_j, i.e., $u_j = f(e_j)$, where f is an affine function, and $j \in \{0, \ldots, n-1\}$.

Proposition 2 *The solution u_j to the full-state space LQT problem (3.22)-(3.24) has a linear expression in terms of e_j.*

Proof. Because u_j is linear in λ_{j+1} in (3.30), an affine relation between λ_j and e_j is sought. If an affine relation between λ_j and e_j can be found, i.e.,

$$\lambda_j = \Phi_j e_j + \Psi_j, \tag{3.32}$$

for some $(m+1) \times (m+1)$ matrix Φ_j and $(m+1) \times 1$ vector Ψ_j, $j \in \{0, \ldots, n-1\}$, then u_j is linear in e_{j+1}. The proof of (3.32) follows using backwards induction on j.

When $j = n$, (3.32) holds since $\Phi_n = F$ and $\Psi_n = 0$ from (3.29). Assume (3.32) is true for $j \in \{i+1, \ldots, n\}$, and now, show that it is also valid for $j = i$. By (3.28) and (3.32),

$$\lambda_i = \lambda_{i+1} + Q_i e_i = \Phi_{i+1} e_{i+1} + \Psi_{i+1} + Q_i e_i. \tag{3.33}$$

Applying (3.30) and (3.32) in (3.31), yields

$$\begin{aligned} e_{i+1} &= e_i + a_i[1/2 - R_i^{-1} a_i^T \lambda_{i+1}] \\ &= e_i + a_i[1/2 - R_i^{-1} a_i^T (\Phi_{i+1} e_{i+1} + \Psi_{i+1})] \end{aligned}$$

$$\Rightarrow e_{i+1} = (I + a_i R_i^{-1} a_i^T \Phi_{i+1})^{-1} \left(e_i + \frac{1}{2} a_i - a_i R_i^{-1} a_i^T \Psi_{i+1} \right). \tag{3.34}$$

Replace e_{i+1} in (3.33) with (3.34), yielding $\lambda_i = \Phi_i e_i + \Psi_i$, with

$$\begin{aligned} \Phi_i =& \Phi_{i+1}(I + a_i R_i^{-1} a_i^T \Phi_{i+1})^{-1} + Q_i \\ \Psi_i =& \Phi_{i+1}(I + a_i R_i^{-1} a_i^T \Phi_{i+1})^{-1} \left(\frac{1}{2} a_i - a_i R_i^{-1} a_i^T \Psi_{i+1} \right) \\ =& - \Phi_{i+1}(I + a_i R_i^{-1} a_i^T \Phi_{i+1})^{-1} a_i R_i^{-1} a_i^T \Psi_{i+1} \\ &+ \Phi_{i+1}(I + a_i R_i^{-1} a_i^T \Phi_{i+1})^{-1} \frac{1}{2} a_i. \end{aligned}$$

Thus, (3.32) is true for $j = i$.

Finally, from (3.30), (3.32) and (3.34), it follows

$$\begin{aligned} u_j &= \frac{1}{2} - R_j^{-1} a_j^T \lambda_{j+1} \\ &= \frac{1}{2} - R_j^{-1} a_j^T \left(\Phi_{j+1} e_{j+1} + \Psi_{j+1}\right) \\ &= \frac{1}{2} - R_j^{-1} a_j^T \left[\Phi_{j+1}(I + a_j R_j^{-1} a_j^T \Phi_{j+1})^{-1}(e_j + \frac{1}{2} a_j - a_j R_j^{-1} a_j^T \Psi_{j+1}) + \Psi_{j+1}\right] \end{aligned}$$

for $k \in \{0, \cdots, n-1\}$, which shows that u_j is linear in e_j and completes the proof. ■

Similarly as in (3.25), the approximate binary solution to the original BIP problem (3.1)-(3.3) is as follows:

$$u_j^* = \begin{cases} 0 \text{ for } u_j < 0.5 \\ 1 \text{ for } u_j \geq 0.5 \end{cases} \tag{3.35}$$

for $j = 0, 1, \ldots, n-1$.

3.3 Numerical Result

3.3.1 Test Problems

The performance of the algorithm on some test problems obtained from MIPLIB [31] are explored. MIPLIB is a standard and widely used benchmark for comparing the performance of various mixed integer programming algorithms, and most of the problems in the MIPLIB arise from real-world applications. Six tests are presented in the numerical tests, including *enigma*, *air*01, *air*03, *air*04, *air*05 and *nw*04. For problem *enigma*, the right-hand-side elements ($\tilde{b}_i, i = 1, \ldots, 21$), the cost coefficients ($\tilde{c}_j, j = 0, \ldots, 99$), the transpose of the coefficient matrix ($\tilde{a}_{ij}, i = 1, \ldots, 21, j = 0, \ldots, 99$) and the optimal solution ($u_j^*, j = 0, \ldots, 99$) are shown in Figure 3.2. Problems *air*01, *air*03, *air*04, *air*05 and *nw*04 are airline crew scheduling type problems. Their right-hand-side elements are all ones, and the coefficient matrices are all binary values. The cost coefficients of all the six problems are positive integer values in the range of 0 to $13,746$. The dimensions for the test problems, the density of the coefficient matrix, the number of 1's and 0's in the optimal solutions and the optimal objective function values are shown in Table 3.1.

Each test problem was solved using branch-and-cut with CPLEX and branch-and-bound in MATLAB. The CPU time (in seconds) for a single run with CPLEX and MATLAB is

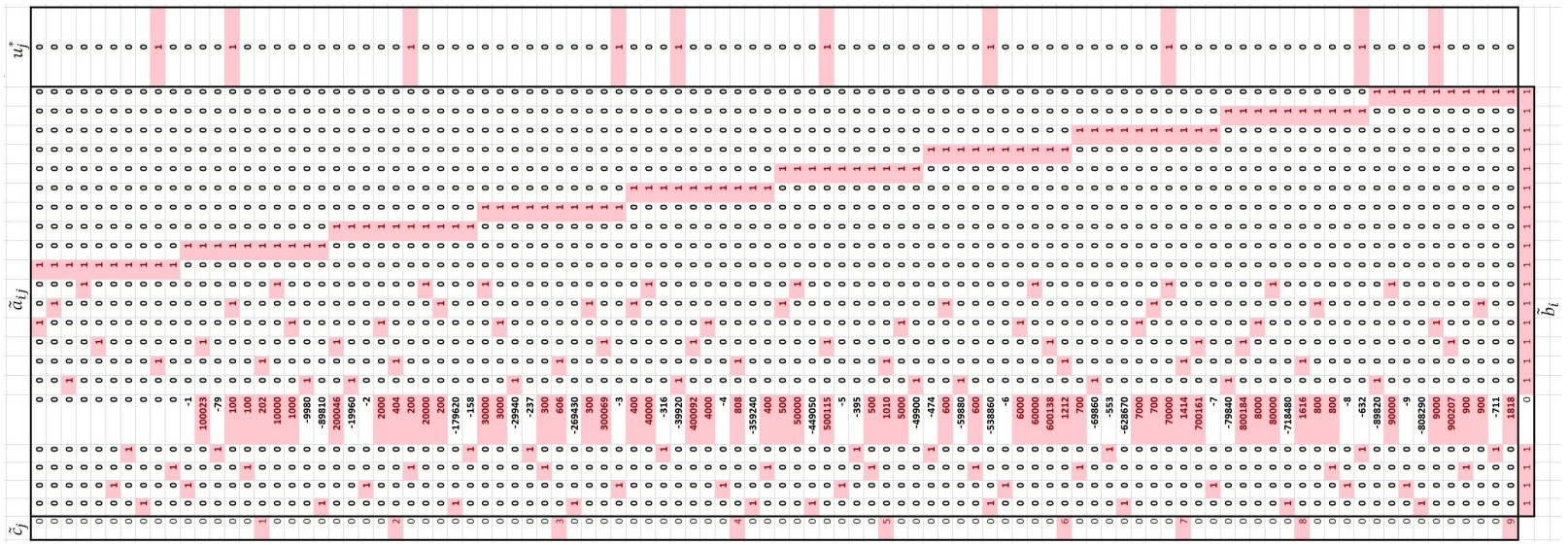

Figure 3.2: Cost coefficients, transpose of coefficient matrix, the right-hand-side elements and the optimal solution of problem *enigma*

also given in Table 3.1. Three of the Matlab problems could not run and got an error message, "the size of the problem is too large for the current solver." All tests are done on an Intel(R) Core(TM) i3 CPU @2.4GHz machine under 64bit Windows7 with 4GB RAM.

Table 3.1: Test problems from MIPLIB

Problem	n	m	Density of coefficient matrix	Number of ones in solution	Optimal objective function value	Time(sec) with branch-and-cut in CPLEX	Time(sec) with branch-and-bound in MATLAB
enigma	100	21	13.76%	10	0	0.23	4.02
air01	771	23	23.77%	6	6,796	0.28	2.86
air03	10,757	124	6.82%	41	340,160	1.05	17.64
air04	8,904	823	1.00%	102	56,137	34.35	too large to run
air05	7,195	426	1.70%	66	26,374	26.66	too large to run
nw04	87,482	36	20.22%	9	16,862	9.83	too large to run

3.3.2 Algorithm Performance Metrics

In the numerical studies, the quality of the approximating solutions are evaluated by the optimality measure, the feasibility measure, the number of constraints violated, the number of 0/1 in the approximate solution and the CPU time (in seconds). The optimality measure is defined as $\frac{\hat{f}-f^*}{f_W-f^*}$ [2], where f^* denotes the true objective function value, $\hat{f}$ denotes the function value found by the proposed method and f_W denotes the worst (largest) function value. The feasibility measure is the summation of the absolute differences of feasibility over all constraints. The CPU time is for a single run with the proposed discrete LQT method in MATLAB.

3.3.3 Numerical Tests

In the numerical tests, different values for parameters Q_j, R_j and F have been experimented on the small problems *enigma* and $air01$. The diagonal elements of Q_j are set to 0, 1 and

Table 3.2: Test results for the discrete LQT-based formulation method

Problem	n	m	Feasibility measure	Number of violated constraints	Number of ones in solution	Optimality measure	CPU time (sec)
enigma	100	21	18	18	1	0	0.03
air01	771	23	13	13	7	2.55%	0.22
air03	10,757	124	138	42	55	-11.68%	34.00
air04	8,904	823	706	522	141	1.43%	3,231.5
air05	7,195	426	322	252	71	-0.55%	698.6
nw04	87,482	36	19	19	13	1.36%	37.9

10, and the results indicate that smaller values are better, so results with $Q_j = 0$ are reported in Table 3.2. Test values for parameter R_j were set to 1, 10, 100 and $1,000$, and there was not much difference in performance on *enigma* and *air*01, so R_j is set to 10 for all tests reported in Table 3.2. As for parameter F, the results show that bigger values are better, so the diagonal elements of F are set to $100,000$. The parameters Q_j penalize the intermediate error values whereas the parameter F penalizes the terminal error at n. Since the terminal error better reflects the original BIP optimality and feasibility measures, intuitively, it makes sense to set $Q_j = 0$ and F large. Values for the weighting factor ω, ranged between 0.5 to 0.9 in the exploratory tests, and the best results were typically for ω between 0.5 and 0.6.

To search for a feasible solution, we ignored the cost coefficients by setting the first element of F to zero, and keeping $Q_j = 0$. The values of the other parameters were the same as before. The new test results for feasibility only are shown in Table 3.3. Comparing with the results shown in Table 3.2, the feasibility measure in Table 3.3 has been improved.

Table 3.1, Table 3.2 and Table 3.3 show that, branch-and-cut in CPLEX ran very quickly and always found an optimal solution; branch-and-bound in MATLAB was slower and only found a feasible solution for *enigma*, *air*01 and *air*03; the proposed discrete LQT

Table 3.3: Test results for feasibility only with the first element of F set to zero

Problem	n	m	Feasibility measure	Number of violated constraints	Number of ones in solution
enigma	100	21	18	18	1
air01	771	23	3	3	12
air03	10,757	124	3	3	103
air04	8,904	823	536	471	110
air05	7,195	426	228	201	60
nw04	87,482	36	10	10	22

method in MATLAB ran slower than branch-and-cut in CPLEX, but generally faster than branch-and-bound in MATLAB. The reason for the slow speed of our algorithm is due to the operations of store and load of the temporary matrices. Even though the numerical results of the proposed method are "worse" than the results of branch-and-cut in CPLEX, the proposed discrete LQT-based formulation method has a potential for extension with polynomial computational complexity.

3.4 Summary and Conclusion

The discrete LQT-based approximation procedure for solving large-scale BIP problems shows much promise because the computational complexity is linear in n (the number of variables) and polynomial in m (the number of constraints), specifically on the order of $O(nm^3)$.

Theorem 1 proves the existence of an optimal binary solution to the LQT problem, which suggests an algorithmic approach for solving the discrete LQT problem.

Numerical results with experimentally chosen parameter values are provided to demonstrate the effectiveness of the proposed approach. The sensitivity of the proposed approximation algorithm to parameter design is also addressed.

Chapter 4

STOCHASTIC DECOMPOSITION APPROACH

In Chapter 3, the discrete LQT-based approximating method is demonstrated to provide advantages in reducing computation while generating a good approximate solution to original BIP problem. The LQT formulation classically requires the solution of a Ricatti equation of large dimension. The computational complexity for solving the LQT problem is polynomial in the time horizon, the dimension of the state space and the number of control variables. In this chapter, a stochastic decomposition algorithm is proposed to further reduce the complexity of the problem by creating approximation subproblems. The error introduced by the partitioning can be modeled by random variables with Gaussian distributions, and then a Kalman filter propagator is used to correct the error and obtain an optimal estimate. The experience with decomposing the LQT problem for the BIP can be generalized to any LQT problem with large state space.

4.1 Stochastic Decomposition Approach for Approximating BIP Problems

4.1.1 Construct two LQGT subproblems

In the full-state space LQT problem (3.22)-(3.24), the time horizon is n, the number of control variables is 1 and the dimension of the state space is related to the number of constraints m. The computational complexity for solving this LQT problem is on the order of $O(nm^3)$. Motivated by Aoki's partitioning method [3], the proposed method to solving the full-state space LQT problem (3.22)-(3.24) is to create two LQGT subproblems with reduced state space dimensionality. In order to define the two subproblems, partition the $m+1$ constraints in (3.23) into one group with s constraints and one group with $t = m - s$ constraints (in the numerical experiments, $s = \lfloor (m+1)/2 \rfloor$). The dynamics, similar to

(3.23), can be rewritten as:

$$e^s_{j+1} = e^s_j + a^s_j u_j + \omega^s_j \tag{4.1}$$

$$e^t_{j+1} = e^t_j + a^t_j u_j + \omega^t_j \tag{4.2}$$

where e^s_j and a^s_j are $(s+1)\times 1$ vectors, and e^t_j and a^t_j are $t\times 1$ vectors. To account for the error due to the reduction of the state space, introduce $(s+1)\times 1$ and $t\times 1$ stochastic vectors, ω^s_j and ω^t_j, for $j = 0, \ldots, n-1$, that are assumed to be zero mean Gaussian distributed variables and independent. The initial conditions are given as, $e^s_0 = -b^s$ where $b^s = [0, \tilde{b}_1, \ldots, \tilde{b}_s]^T$ and $e^t_0 = -b^t$ where $b^t = [\tilde{b}_{s+1}, \ldots, \tilde{b}_m]^T$.

To link the two subproblems, define observation equations using information from a previous iteration. Let l denote the iteration and suppose both s and t subproblems have been solved in iteration $l-1$. Then for the l^{th} iteration, the observation equation for subproblem s uses the control from subproblem s in iteration $l-1$, denoted $u^{s(l-1)}_j$, and the state from subproblem t in iteration $l-1$, denoted $\hat{e}^{t(l-1)}_j$. Let $z^{s(l-1)}_j$ represent the observation on the $l-1$ iteration, and rewrite (4.2) to get

$$\hat{e}^{t(l-1)}_{j+1} = \hat{e}^{t(l-1)}_j + a^t_j u^{s(l-1)}_j + \omega^t_j. \tag{4.3}$$

Use the following linear form for the control,

$$u^{s(l-1)}_j = c^{s(l-1)}_j \hat{e}^{s(l-1)}_j + d^{s(l-1)}_j, \tag{4.4}$$

where $c^{s(l-1)}_j$ is a $1\times(s+1)$ vector and $d^{s(l-1)}_j$ is a scalar, and substitute $u^{s(l-1)}_j$ into (4.3) to obtain

$$\hat{e}^{t(l-1)}_{j+1} = \hat{e}^{t(l-1)}_j + a^t_j \left(c^{s(l-1)}_j \hat{e}^{s(l-1)}_j + d^{s(l-1)}_j \right) + \omega^t_j. \tag{4.5}$$

Now, to obtain the observation equation, replace $\hat{e}^{s(l-1)}_j$ with the state variables $e^{s(l)}_j$ and $z^{s(l-1)}_j$, the observed value, is:

$$z^{s(l-1)}_j = \hat{e}^{t(l-1)}_{j+1} - \hat{e}^{t(l-1)}_j - a^t_j d^{s(l-1)}_j = a^t_j c^{s(l-1)}_j e^{s(l)}_j + \omega^t_j. \tag{4.6}$$

To summarize, the LQGT subproblem s on the l^{th} iteration, $P_s^{(l)}$, is:

$$\min_{\substack{u_j \\ j=0,\ldots,n-1}} \quad \mathbb{E}\left[\frac{1}{2}\sum_{j=0}^{n-1}\left((e_j^{s(l)})^T Q_j^s e_j^{s(l)} + u_j(u_j-1)R_j\right) + \frac{1}{2}(e_n^{s(l)})^T F^s e_n^{s(l)} \middle| \hat{e}_j^{t(l-1)}, c_j^{s(l-1)}, d_j^{s(l-1)}\right]$$

$$\text{s.t.} \quad e_{j+1}^{s(l)} = e_j^{s(l)} + a_j^s u_j + \omega_j^s \qquad j = 0,\ldots,n-1 \tag{4.7}$$

$$z_j^{s(l-1)} = \hat{e}_{j+1}^{t(l-1)} - \hat{e}_j^{t(l-1)} - a_j^t d_j^{s(l-1)} = a_j^t c_j^{s(l-1)} e_j^{s(l)} + \omega_j^t \quad j = 0,\ldots,n-1 \tag{4.8}$$

$$e_0^{s(l)} = -b^s. \tag{4.9}$$

The parameters Q_j^s and F^s have the same forms as Q_j and F in the full-state space LQT problem (3.22)-(3.24) except the dimensions have been changed accordingly.

The observation equation for subproblem t on the l^{th} iteration uses the control from subproblem t in iteration $l-1$, denoted $u_j^{t(l-1)}$, and the state from subproblem s in iteration l, denoted $\hat{e}_j^{s(l)}$. Again, the control has a linear form,

$$u_j^{t(l-1)} = c_j^{t(l-1)} \hat{e}_j^{t(l-1)} + d_j^{t(l-1)}, \tag{4.10}$$

where $c_j^{t(l-1)}$ is a $1 \times (t+1)$ vector and $d_j^{t(l-1)}$ is a scalar. The observation equation is:

$$z_j^{t(l-1)} = \hat{e}_{j+1}^{s(l)} - \hat{e}_j^{s(l)} - a_j^s d_j^{t(l-1)} = a_j^s c_j^{t(l-1)} e_j^{t(l)} + \omega_j^s. \tag{4.11}$$

To summarize, the LQGT subproblem t on the l^{th} iteration, $P_t^{(l)}$, is:

$$\min_{\substack{u_j \\ j=0,\ldots,n-1}} \quad \mathbb{E}\left[\frac{1}{2}\sum_{j=0}^{n-1}\left((e_j^{t(l)})^T Q_j^t e_j^{t(l)} + u_j(u_j-1)R_j\right) + \frac{1}{2}(e_n^{t(l)})^T F^t e_n^{t(l)} \middle| \hat{e}_j^{s(l-1)}, c_j^{t(l-1)}, d_j^{t(l-1)}\right]$$

$$\text{s.t.} \quad e_{j+1}^{t(l)} = e_j^{t(l)} + a_j^t u_j + \omega_j^t \qquad j = 0,\ldots,n-1 \tag{4.12}$$

$$z_j^{t(l-1)} = \hat{e}_{j+1}^{s(l-1)} - \hat{e}_j^{s(l-1)} - a_j^s d_j^{t(l-1)} = a_j^s c_j^{t(l-1)} e_j^{t(l)} + \omega_j^s \quad j = 0,\ldots,n-1 \tag{4.13}$$

$$e_0^{t(l)} = -b^t. \tag{4.14}$$

The parameters Q_j^t and F^t have the same forms as Q_j and F in the full-state space LQT problem (3.22)-(3.24) except the dimensions have been changed accordingly.

Both the LQGT subproblems can be solved deterministically using the separation principle [8, 12]. First solve $P_s^{(l)}$ and $P_t^{(l)}$ deterministically by the dynamic programming method [6] and then obtain the optimal control $u_j^{s(l)}$, $u_j^{t(l)}$ and the estimate state $\hat{e}_j^{s(l)}$, $\hat{e}_j^{t(l)}$ respectively via the Kalman filtering technique [12]. The flowchart of the stochastic decomposition approach for approximating original BIP solutions is shown in Figure 4.1.

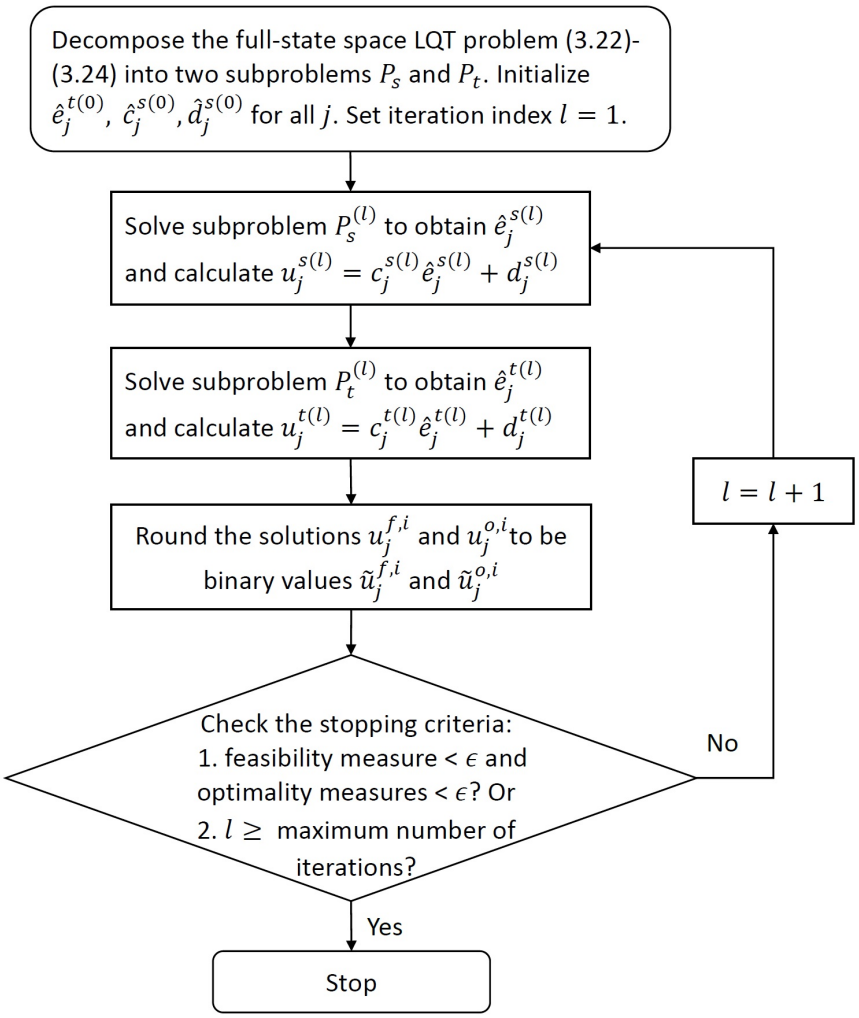

Figure 4.1: Flowchart of the stochastic decomposition approach for approximating BIP

4.1.2 Discussion of the decomposition method

In this section, a comparison of the solution for the full-state space LQT problem (3.22)-(3.24) and the solutions for the decomposed LQGT subproblem, $P_s^{(l)}$ and $P_t^{(l)}$, is discussed, for a large iteration index l. Following [12], the linear form for the optimal control u_j^* for the full-state space LQT problem (3.22)-(3.24) can be written as $u_j^* = c_j e_j^* + d_j$, for $j = n-1, \ldots, 1, 0$, where

$$c_j = -\frac{a_j^T S_{j+1}}{R_j + a_j^T S_{j+1} a_j}, \quad d_j = \frac{(R_j/2) - a_j^T T_{j+1}}{R_j + a_j^T S_{j+1} a_j}, \tag{4.15}$$

$$S_j = Q_j + c_j^T R_j c_j + (I + a_j c_j)^T S_{j+1}(I + a_j c_j), \tag{4.16}$$

$$T_j = c_j^T d_j R_j + (I + a_j c_j)^T S_{j+1} a_j d_j + (I + a_j c_j)^T T_{j+1} - c_j^T (R_j/2), \tag{4.17}$$

with $S_n = F, T_n = 0$. The optimal state can be written as:

$$e_{j+1}^* = e_j^* + a_j(c_j e_j^* + d_j) = (I + a_j c_j) e_j^* + a_j d_j. \tag{4.18}$$

And for comparison purposes, write e_j^* as $[e_j^{s*}\ e_j^{t*}]^T$,

$$\begin{aligned} \begin{bmatrix} e_{j+1}^{s*} \\ e_{j+1}^{t*} \end{bmatrix} &= \left(I + \begin{bmatrix} a_j^s \\ a_j^t \end{bmatrix} \begin{bmatrix} c_j^s & c_j^t \end{bmatrix} \right) \begin{bmatrix} e_j^{s*} \\ e_j^{t*} \end{bmatrix} + \begin{bmatrix} a_j^s d_j \\ a_j^t d_j \end{bmatrix} \\ &= \begin{bmatrix} I + a_j^s c_j^s & a_j^s c_j^t \\ a_j^t c_j^s & I + a_j^t c_j^t \end{bmatrix} \begin{bmatrix} e_j^{s*} \\ e_j^{t*} \end{bmatrix} + \begin{bmatrix} a_j^s d_j \\ a_j^t d_j \end{bmatrix} \end{aligned} \tag{4.19}$$

For the subproblem $P_s^{(l)}$, let $\hat{e}_{j+1|j}^{s(l)}$ be the estimate of the state $e_{j+1}^{s(l)}$ before the Kalman correction and $\hat{e}_{j+1|j+1}^{s(l)}$ be the estimate of the state $e_{j+1}^{s(l)}$ after the Kalman correction. Then from (4.7), (4.8) and (4.4),

$$\hat{e}_{j+1|j}^{s(l)} = \hat{e}_{j|j}^{s(l)} + a_j^s u_j^{s(l)} = (I + a_j^s c_j^{s(l)}) \hat{e}_{j|j}^{s(l)} + a_j^s d_j^{s(l)} \tag{4.20}$$

$$\hat{e}_{j+1|j+1}^{s(l)} = \hat{e}_{j+1|j}^{s(l)} + G_{j+1}^{s(l)} \left(z_{j+1}^{s(l-1)} - a_{j+1}^t c_{j+1}^{s(l-1)} \hat{e}_{j+1|j}^{s(l)} \right) \tag{4.21}$$

$$= \left(I - G_{j+1}^{s(l)} a_{j+1}^t c_{j+1}^{s(l-1)} \right) \hat{e}_{j+1|j}^{s(l)} + G_{j+1}^{s(l)} z_{j+1}^{s(l-1)} \tag{4.22}$$

$$= \left(I - G_{j+1}^{s(l)} a_{j+1}^t c_{j+1}^{s(l-1)} \right) \left[\left(I + a_j^s c_j^{s(l)} \right) \hat{e}_{j|j}^{s(l)} + a_j^s d_j^{s(l)} \right] + G_{j+1}^{s(l)} z_{j+1}^{s(l-1)} \tag{4.23}$$

$$= M_j^{s(l)} \hat{e}_{j|j}^{s(l)} + N_j^{s(l)} \tag{4.24}$$

where

$$M_j^{s(l)} = \left(I - G_{j+1}^{s(l)} a_{j+1}^t c_{j+1}^{s(l-1)} \right) \left(I + a_j^s c_j^{s(l)} \right) \tag{4.25}$$

$$N_j^{s(l)} = \left(I - G_{j+1}^{s(l)} a_{j+1}^t c_{j+1}^{s(l-1)} \right) a_j^s d_j^{s(l)} + G_{j+1}^{s(l)} z_{j+1}^{s(l-1)} \tag{4.26}$$

and the Kalman gains $G_j^{s(l)}$, for $j = 0, \cdots, n-1$, can be calculated recursively as follows [12]:

$$\begin{cases} P_{0,0}^{s(l)} &= cov[\omega_0^{s(l)}] = I, \\ P_{j|j-1}^{s(l)} &= P_{j-1|j-1}^{s(l)} + cov[\omega_{j-1}^{s(l)}], \\ G_j^{s(l)} &= P_{j|j-1}^{s(l)} \left(a_j^t c_j^{s(l-1)}\right)^T \left[a_j^t c_j^{s(l-1)} P_{j|j-1}^{s(l)} \left(a_j^t c_j^{s(l-1)}\right)^T + cov[\omega_{j-1}^{t(l)}]\right]^{-1}, \\ P_{j|j}^{s(l)} &= \left(I - G_j^{s(l)} a_j^t c_j^{s(l-1)}\right) P_{j|j-1}^{s(l)}. \end{cases} \tag{4.27}$$

For comparison purposes, write $\hat{e}_{j|j}^{s(l)}$ and $\hat{e}_{j|j}^{t(l)}$ in a vector form, yielding

$$\begin{bmatrix} \hat{e}_{j+1|j+1}^{s(l)} \\ \hat{e}_{j+1|j+1}^{t(l)} \end{bmatrix} = \begin{bmatrix} M_j^{s(l)} \hat{e}_{j|j}^{s(l)} + N_j^{s(l)} \\ M_j^{t(l)} \hat{e}_{j|j}^{t(l)} + N_j^{t(l)} \end{bmatrix} = \begin{bmatrix} M_j^{s(l)} & 0 \\ 0 & M_j^{t(l)} \end{bmatrix} \begin{bmatrix} \hat{e}_{j|j}^{s(l)} \\ \hat{e}_{j|j}^{t(l)} \end{bmatrix} + \begin{bmatrix} N_j^{s(l)} \\ N_j^{t(l)} \end{bmatrix}. \tag{4.28}$$

Let $\begin{bmatrix} r_j^{s(l)} \\ r_j^{t(l)} \end{bmatrix} = \begin{bmatrix} e_j^{s*} - \hat{e}_{j|j}^{s(l)} \\ e_j^{t*} - \hat{e}_{j|j}^{t(l)} \end{bmatrix}$ to represent the difference between the full-state space solution and the solution to the decomposition method on the l^{th} iteration. Subtracting (4.28) from (4.19) and expressing $\begin{bmatrix} e_j^{s*} & e_j^{t*} \end{bmatrix}^T = \begin{bmatrix} r_j^{s(l)} + \hat{e}_{j|j}^{s(l)} & r_j^{t(l)} + \hat{e}_{j|j}^{t(l)} \end{bmatrix}^T$, yields

$$\begin{aligned} \begin{bmatrix} r_{j+1}^{s(l)} \\ r_{j+1}^{t(l)} \end{bmatrix} &= \begin{bmatrix} (I + a_j^s c_j^s)(\hat{e}_{j|j}^{s(l)} + r_j^{s(l)}) + a_j^s c_j^t (\hat{e}_{j|j}^{t(l)} + r_j^{t(l)}) - M_j^{s(l)} \hat{e}_{j|j}^{s(l)} + (a_j^s d_j - N_j^{s(l)}) \\ a_j^t c_j^s (\hat{e}_{j|j}^{s(l)} + r_j^{s(l)}) + (I + a_j^t c_j^t)(\hat{e}_{j|j}^{t(l)} + r_j^{t(l)}) - M_j^{t(l)} \hat{e}_{j|j}^{t(l)} + (a_j^t d_j - N_j^{t(l)}) \end{bmatrix} \\ &= \begin{bmatrix} I + a_j^s c_j^s & a_j^s c_j^t \\ a_j^t c_j^s & I + a_j^t c_j^t \end{bmatrix} \begin{bmatrix} r_j^{s(l)} \\ r_j^{t(l)} \end{bmatrix} \\ &\quad + \begin{bmatrix} I + a_j^s c_j^s - M_j^{s(l)} & a_j^s c_j^t \\ a_j^t c_j^s & I + a_j^t c_j^t - M_j^{t(l)} \end{bmatrix} \begin{bmatrix} \hat{e}_{j|j}^{s(l)} \\ \hat{e}_{j|j}^{t(l)} \end{bmatrix} + \begin{bmatrix} a_j^s d_j - N_j^{s(l)} \\ a_j^t d_j - N_j^{t(l)} \end{bmatrix}. \end{aligned} \tag{4.29}$$

When iteration l is large, the Kalman gains $G_j^{s(l)}$ and $G_j^{t(l)}$ are small, hence $M_j^{s(l)} \approx I + a_j^s c_j^{s(l)}, N_j^{s(l)} \approx a_j^s d_j^{s(l)}, M_j^{t(l)} \approx I + a_j^t c_j^{t(l)}$ and $N_j^{t(l)} \approx a_j^t d_j^{t(l)}$. For large l, (4.29) yields

$$\begin{aligned} \begin{bmatrix} r_{j+1}^{s(l)} \\ r_{j+1}^{t(l)} \end{bmatrix} &\approx \begin{bmatrix} I + a_j^s c_j^s & a_j^s c_j^t \\ a_j^t c_j^s & I + a_j^t c_j^t \end{bmatrix} \begin{bmatrix} r_j^{s(l)} \\ r_j^{t(l)} \end{bmatrix} \\ &\quad + \begin{bmatrix} a_j^s (c_j^s - c_j^{s(l)}) & a_j^s c_j^t \\ a_j^t c_j^s & a_j^t (c_j^t - c_j^{t(l)}) \end{bmatrix} \begin{bmatrix} \hat{e}_{j|j}^{s(l)} \\ \hat{e}_{j|j}^{t(l)} \end{bmatrix} + \begin{bmatrix} a_j^s (d_j - d_j^{s(l)}) \\ a_j^t (d_j - d_j^{t(l)}) \end{bmatrix}. \end{aligned} \tag{4.30}$$

In (4.30), it is observed that if the eigenvalues of the product matrices involving

$$\begin{bmatrix} I + a_j^s c_j^s & a_j^s c_j^t \\ a_j^t c_j^s & I + a_j^t c_j^t \end{bmatrix}$$

with propagation of r have magnitude less than one, and the driving terms

$$\begin{bmatrix} a_j^s (d_j - d_j^{s(l)}) \\ a_j^t (d_j - d_j^{t(l)}) \end{bmatrix} + \begin{bmatrix} a_j^s (c_j^s - c_j^{s(l)}) & a_j^s c_j^t \\ a_j^t c_j^s & a_j^t (c_j^t - c_j^{t(l)}) \end{bmatrix} \begin{bmatrix} \hat{e}_{j|j}^{s(l)} \\ \hat{e}_{j|j}^{t(l)} \end{bmatrix}$$

are small, then (4.30) is a contraction mapping, and $\begin{bmatrix} r_j^{s(l)} & r_j^{t(l)} \end{bmatrix}^T$ will be small when j goes large. Thus for large l and large j, the difference between the full state solution and decomposed solution is small, especially the last state vector $\begin{bmatrix} r_n^{s(l)} & r_n^{t(l)} \end{bmatrix}^T$ is small for large n. Additionally, $\begin{bmatrix} r_0^{s(l)} & r_0^{t(l)} \end{bmatrix}^T = 0$ because the initial state vectors for the respective problems are determined by $-b$.

4.1.3 Numerical Results

The algorithm in Figure 4.1 is explored with six test problems *enigma*, *air*01, *air*03, *air*04, *air*05 and *nw*04 as used in Chapter 3. The feasibility measure and the optimality measure are the same as in Chapter 3. All tests are done on a same Intel(R) Core(TM) i3 CPU @2.4GHz machine under 64bit Windows7 with 4GB RAM as in Chapter 3. In the numerical tests in Table 4.1, we set $s = \lfloor 0.5(m+1 \rfloor)$, $Q_j = 0$, $R_j = 10$, and the diagonal elements of F to $100,000$. We also tested the covariance matrices of the white Gaussian noise terms ω_j^s and ω_j^t as $0.5I$,I, $50I$ and $100I$, the results with the best combinations of Q_j, R_j, F and covariance values are reported.

Numerical results show that the proposed stochastic decomposition approach is effective for approximately solving large-scale BIP problems. Comparing the results in Table 3.2, the feasibility and optimality measures are comparable, but the total computational time with the stochastic decomposition method is longer than that with the discrete LQT-based formulation method proposed in Chapter 3. The reason is that the stochastic decomposition method is based on a reduced set of constraints and is solved iteratively, while the discrete LQT-based method is based on the full-state space and is solved only once. Therefore,

Table 4.1: Test results for the proposed stochastic decomposition method in Figure 4.1

Problem	Algorithm in Figure 3.1					Algorithm in Figure 4.1				
	Feasibility measure	#violated constraints	#ones in soln	Optimality measure	CPU time (sec)	Feasibility measure	#violated constraints	#ones in soln	Optimality measure	CPU time (sec)
enigma	18	18	1	0%	0.03	17	17	1	0%	0.19
air01	13	13	7	2.55%	0.22	9	9	8	-2.05%	1.31
air03	138	42	55	-11.68%	34.00	94	31	40	-2.46%	134.6
air04	706	522	141	1.43%	3231.5	729	473	211	-3.96%	8735.1
air05	322	252	71	-0.55%	698.7	348	286	94	-47.67%	1652.1
nw04	19	19	13	1.36%	37.9	46	46	17	-4.94%	202.5

we recommend the decomposition method only when necessary for BIPs with a very large number of constraints.

4.2 Stochastic Decomposition Approach for General Large LQT problems

The stochastic decomposition approach was motivated by the BIP problem with a large number of constraints. But the approach can also be extended to a general LQT problem. Consider a large scale LQT problem, P_0, of the form

$$\min_{\substack{u_j \\ j=0,\ldots,N-1}} \quad \sum_{j=0}^{N-1} \left(x_j^T Q_j x_j + u_j^T R_j u_j\right) + (x_N - x_F)^T F (x_n - x_F) \tag{4.31}$$

$$\text{s.t.} \qquad x_{j+1} = A_j x_j + B_j u_j \qquad \text{for} \quad j = 0, 1, \ldots, N-1 \tag{4.32}$$

where $x_j \in \mathbb{R}^n$ is the state variable for $j = 0, 1, \ldots, N$, with x_0 given, and $u_j \in \mathbb{R}^m$ is the control variable for $j = 0, 1, \ldots, N-1$. In the quadratic criterion (4.31), $x_F \in \mathbb{R}^n$ is the known target state, the cost parameter matrices $Q_j \in \mathbb{R}^{n \times n}$ and $F \in \mathbb{R}^{n \times n}$ are positive semi-definite ($Q_j \succeq 0, F \succeq 0$) and $R_j \in \mathbb{R}^{m \times m}$ is positive definite ($R_j \succ 0$). In the dynamics equation (4.32), $A_j \in \mathbb{R}^{n \times n}$, $B_j \in \mathbb{R}^{n \times m}$ and the linear system is assumed to be controllable for almost all k. In the following sections, cases with n and N very large, and $m = 1$ are considered.

4.2.1 Partition the Problem in State Space

From the prospective of an LQT problem, partitioning BIP by constraints means to partition in the state space for P_0. The deterministic LQT problem P_0 with $m = 1$ can be solved classically by dynamic programming approach [15], however, the computational complexity grows on the order of $O(Nn^3)$ which is impractical for large n and N. Aoki [3] suggested a partitioning scheme to "decouple by stochasticity". In this section, this concept is extended to motivate a practical algorithm for large n that "divides and conquers." Whereas Aoki's partitioning scheme uses the differences in eigenvalues to create the partition, the proposed extension allows the eigenvalues to be in the same range and still partition effectively to achieve a reduced computational complexity.

Starting by partitioning the state vector into two blocks and rewriting the dynamics in (4.32), yields

$$\begin{bmatrix} x_{j+1}^1 \\ x_{j+1}^2 \end{bmatrix} = \begin{bmatrix} A_j^{11} & A_j^{12} \\ A_j^{21} & A_j^{22} \end{bmatrix} \begin{bmatrix} x_j^1 \\ x_j^2 \end{bmatrix} + \begin{bmatrix} B_j^1 \\ B_j^2 \end{bmatrix} u_j. \tag{4.33}$$

The idea is to construct two decoupled linear quadratic Gaussian tracking (LQGT) problems and introduce random error to account for the approximation error. Instead of solving the original optimal control problem P_0, a sequence of the two constructed LQGT problems is solved repeatedly to obtain an efficient approximate solution.

This development uses two blocks in the partitioning, however, the idea can be extended to more blocks. In addition, the size of each block and sequencing of the subproblems may vary, but for presentation purposes the development is presented with fixed size and sequencing.

The two blocks in (4.33) are named as active equation and passive equation. Use s as an indicator for the active equation and t for the passive equation, and rewrite (4.33) as follows:

$$x_{j+1}^s = A_j^{ss} x_j^s + A_j^{st} x_j^t + B_j^s u_j, \tag{4.34}$$

$$x_{j+1}^t = A_j^{ts} x_j^s + A_j^{tt} x_j^t + B_j^t u_j. \tag{4.35}$$

Modify the active equation (4.34) to create a new stochastic dynamic equation and treat the passive equation (4.35) as the observation equation to correct the estimation. For the

active equation in (4.34), replace the coupling term $A_j^{st}x_j^t$ with a random error w_j^s and let $\tilde{x}_j$ denote the approximating states. Add l as an iteration index for an outer loop on the sequencing of subproblems, yielding

$$\tilde{x}_{j+1}^{s(l)} = A_j^{ss}\tilde{x}_j^{s(l)} + B_j^s u_j^{(l)} + w_j^{s(l)}. \tag{4.36}$$

The random error $w_j^{s(l)}$ is assumed to be a white Gaussian noise, i.e.,

$$w_j^{s(l)} \sim \mathcal{N}(0, \Theta_j^{s(l)}) \tag{4.37}$$

with

$$\Theta_j^{s(l)} = A_N^{st} x_N^{t(l-1)} [x_N^{t(l-1)}]^T [A_N^{st}]^T + \delta_1^{(l)} I. \tag{4.38}$$

The random error has zero mean because the approximate error is expected to be symmetric, and the covariance matrix $\Theta_j^{s(l)}$ is associated with the error from the previous iteration, $\delta_1^{(l)}$ is a scalar for adjusting the covariance matrix $\Theta_j^{s(l)}$. For the passive equation in (4.35), a random error v_j^t is introduced to replace the term $A_j^{tt}\tilde{x}_j^{t(l)}$ and let $y_j^{t(l)}$ denote $\tilde{x}_{j+1}^{t(l)} - B_j^t u_j^{(l)}$, yielding

$$y_j^{t(l)} = A_j^{ts}\tilde{x}_j^{s(l)} + v_j^{t(l)}. \tag{4.39}$$

The random error $v_j^{t(l)}$ is also assumed to be a white Gaussian noise, i.e.,

$$v_j^{t(l)} \sim \mathcal{N}(0, \Lambda_j^t) \tag{4.40}$$

with

$$\Lambda_j^{t(l)} = \left(\tilde{x}_N^{t(l-1)} - B_{N-1}^t u_{N-1}^{(l-1)}\right)\left(\tilde{x}_N^{t(l-1)} - B_{N-1}^t u_{N-1}^{(l-1)}\right)^T + \delta_2^{(l)} I \tag{4.41}$$

where $\delta_2^{(l)}$ is a scalar for adjusting the covariance matrix $\Lambda_j^{t(l)}$, and $v_j^{t(l)}$ is assumed to be uncorrelated with $w_j^{s(l)}$, i.e., $\mathbb{E}[w_j^{s(l)}(v_j^{t(l)})^T] = 0$.

Now the first stochastic approximation LQGT problem, $P_{st}^{(l)}$, with $s = 1$ and $t = 2$ is created as follows:

$$\min_{\substack{u_j^{(l)} \\ j=0,\ldots,N-1}} \quad \mathbb{E}\left\{ \sum_{j=0}^{N-1} \left([\tilde{x}_j^{s(l)}]^T Q_j \tilde{x}_j^{s(l)} + [u_j^{(l)}]^T R_j u_j^{(l)} \right) + (\tilde{x}_N^{s(l)} - x_F^s)^T F (\tilde{x}_N^{s(l)} - x_F^s) \right\}$$

$$\text{s.t.} \quad \tilde{x}_{j+1}^{s(l)} = A_j^{ss}\tilde{x}_j^{s(l)} + B_j^s u_j^{(l)} + w_j^{s(l)} \tag{4.42}$$

$$y_j^{t(l)} = A_j^{ts}\tilde{x}_{j+1}^{s(l)} + v_j^{t(l)} \tag{4.43}$$

with initial conditions for $l = 1$, $y_j^{t(l)}$ is arbitrary, and $\Theta_j^{s(l)}, \Lambda_j^{t(l)}$ are positive definite arbitrary. For example,

$$\tilde{x}_0^{s(l)} = x_0^s, \quad y_j^{t(l)} = x_0^t, \quad \Theta_j^{s(l)} = I, \quad \Lambda_j^{t(l)} = I. \tag{4.44}$$

The second LQGT problem has the same form as the first one except $s = 2$ and $t = 1$.

To summarize, the stochastic decomposition approach is to partition a deterministic LQT system which is controllable into two stochastic LQGT systems which are stablilizable and detectable, then iteratively solve the two constructed LQGT problems $P_{st}^{(l)}$. First, solve Problem $P_{st}^{(l)}$ with $s = 1$, $t = 2$ and the initial conditions in (4.44), and get the approximate solution of the control sequence $u_j^{*(l)}$ for $j = 0, \ldots, N-1$. Then, solve Problem $P_{st}^{(l)}$ with $s = 2$ and $t = 1$ based on the control $u_j^{*(l)}$ from the previous step, and get the updated control sequence $u_j^{*(l)}$. Next, check whether the control sequence $u_j^{*(l)}$ converges: if it converges, then stop the algorithm; otherwise, let $l = l + 1$ and perform a new iteration for the outer loop.

4.2.2 State Estimate using Kalman Filter

The separation principle [8, 12] shows that for the LQGT problem P_{st}, there exists an optimal feedback control of the form

$$u_j^{*(l)} = -C_j^{(l)} \hat{x}_j^{s(l)} \tag{4.45}$$

with $u_N^{*(l)} = 0$, where $\hat{x}_j^{s(l)}$ is the optimal estimation of the unknown state vector $\tilde{x}_j^{s(l)}$ obtained using the standard Kalman filtering algorithm from the available observation data $y_j^{t(l)}$, and $C_j^{(l)}$ are constant matrices computed from the backward recursive algorithm as follows:

$$\begin{cases} S_N^{(l)} &= F, \\ S_j^{(l)} &= [A_j^{ss}]^T S_{j+1}^{(l)} A_j^{ss} - [A_j^{ss}]^T S_{j+1}^{(l)} B_j^s \big([B_j^s]^T S_{j+1}^{(l)} B_j^s + R_j\big)^{-1} [B_j^s]^T S_{j+1}^{(l)} A_j^{ss} + Q_j, \\ C_j^{(l)} &= \big([B_j^s]^T S_{j+1}^{(l)} B_j^s + R_j\big)^{-1} [B_j^s]^T S_{j+1}^{(l)} A_j^{ss}, \qquad j = N-1, \cdots, 0. \end{cases} \tag{4.46}$$

The Kalman filter algorithm [12] is used to obtain the state estimates $\hat{x}_j^{s(l)}$ needed in

(4.45) by solving the following prediction-correction formula recursively:

$$\text{prediction equation:} \quad \hat{x}_{j+1|j}^{s(l)} = A_j^{ss}\hat{x}_j^{s(l)} + B_j^s u_j^{*(l)}, \tag{4.47}$$

$$\text{correction equation:} \quad \hat{x}_{j+1}^{s(l)} = \hat{x}_{j+1|j}^{s(l)} + G_{j+1}^{s(l)}\left(y_{j+1}^{t(l)} - A_{j+1}^{ts}\hat{x}_{j+1|j}^{s(l)}\right), \tag{4.48}$$

for $j = 0, \cdots, N-1$, and the initial estimate is $\hat{x}_0^{s(l=1)} = x_0^{s(l=1)}$. Given the known optimal estimator $\hat{x}_j^{s(l)}$ at time j, $\hat{x}_{j+1|j}^{s(l)}$ at time $j+1$ in (4.47) can be predicted. Next, the Kalman gains $G_{j+1}^{s(l)}$ are calculated and the correction equation in (4.48) is used to correct the prediction value and get the optimal estimator $\hat{x}_{j+1}^{s(l)}$ at time $j+1$.

The recursive scheme for the Kalman gains $G_j^{s(l)}$ is as follows:

$$\begin{cases} P_0^{(l)} &= cov[x_0^s], \\ P_{j+1|j}^{(l)} &= A_j^{ss}P_j^{(l)}[A_j^{ss}]^T + \Theta_j^{s(l)}, \\ G_{j+1}^{s(l)} &= P_{j+1|j}[A_{j+1}^{ts}]^T\left(A_{j+1}^{ts}P_{j+1|j}^{(l)}[A_{j+1}^{ts}]^T + \Lambda_{j+1}^{t(l)}\right)^{-1}, \\ P_{j+1}^{(l)} &= \left(I - G_{j+1}^{s(l)}A_{j+1}^{ts}\right)P_{j+1|j}^{(l)}, \qquad j = 0, \cdots, N-1. \end{cases} \tag{4.49}$$

This algorithm is a penalty method algorithm [34], and is summarized in the flowchart in Figure 4.2, where a sequence of smaller subproblems are solved iteratively to obtain an approximation to the original full-state space LQT problem. This development is presented with decomposing P_0 into two subproblems, however, the idea can be extended to more than two subproblems. In addition, the size and sequencing of the subproblems may vary.

4.2.3 Convergence Discussion

In this section, the convergence properties of the LQT solution are presented.

Theorem 2 *Given a controllable LQT problem P_0, a partitioning scheme can always be found such that the two constructed LQGT problems $P_{st}^{(l)}$ ($s = 1, t = 2$ and $s = 2, t = 1$) are stabilizable and detectable.*

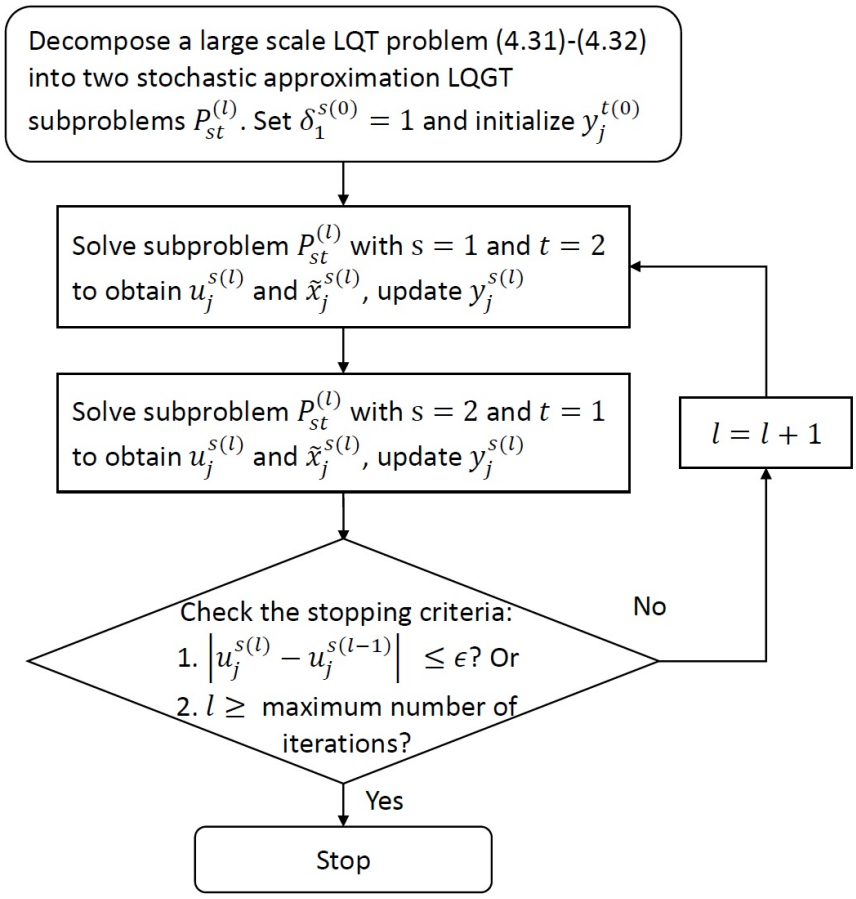

Figure 4.2: Flowchart of the stochastic decomposition approach for general large LQT problems

Proof. The state of the original system at time $1, 2, \ldots, N$ is given by

$$x_1 = A_0 x_0 + B_0 u_0$$

$$x_2 = A_1 x_1 + B_1 u_1 = A_1 A_0 x_0 + A_1 B_0 u_0 + B_1 u_1$$

$$x_3 = A_2 x_2 + B_2 u_2 = A_2 A_1 A_0 x_0 + A_2 A_1 B_0 u_0 + A_2 B_1 u_1 + B_2 u_2$$

$$\vdots$$

$$x_{j+1} = A_j x_j + B_j u_j = \left(\prod_{l=0}^{j} A_l\right) x_0 + \sum_{p=0}^{j-1} \left(\prod_{j=p+1}^{j} A_j\right) B_p u_p + B_j u_j$$

$$\vdots$$

$$x_N = A_{N-1}x_{N-1} + B_{N-1}u_{N-1} = \left(\prod_{l=0}^{N-1} A_l\right) x_0 + \sum_{p=0}^{N-2} \left(\prod_{j=p+1}^{N-1} A_j\right) B_p u_p + B_{N-1}u_{N-1}$$

and in matrix form it is

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{j+1} \\ \vdots \\ x_N \end{bmatrix} - \begin{bmatrix} A_0 \\ A_1A_0 \\ A_2A_1A_0 \\ \vdots \\ \left(\prod_{l=0}^{j} A_l\right) \\ \vdots \\ \left(\prod_{l=0}^{N-1} A_l\right) \end{bmatrix} x_0$$

$$= \begin{bmatrix} B_0 & 0 & 0 & \dots & 0 \\ A_1B_0 & B_1 & 0 & \dots & 0 \\ A_2A_1B_0 & A_2B_1 & B_2 & & 0 \\ \vdots & \vdots & \vdots & \ddots & \\ \left(\prod_{l=1}^{j} A_l\right) B_0 & \left(\prod_{l=2}^{j} A_l\right) B_1 & \left(\prod_{l=3}^{j} A_l\right) B_2 & \dots & B_j \\ \vdots & \vdots & \vdots & \ddots & \\ \left(\prod_{l=1}^{N-1} A_l\right) B_0 & \left(\prod_{l=2}^{N-1} A_l\right) B_1 & \left(\prod_{l=3}^{N-1} A_l\right) B_2 & \dots & B_{N-1} \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_j \\ \vdots \\ u_{N-1} \end{bmatrix}.$$

Let $\mathbf{M} = \begin{bmatrix} \left(\prod_{l=1}^{N-1} A_l\right) B_0 & \left(\prod_{l=2}^{N-1} A_l\right) B_1 & \left(\prod_{l=3}^{N-1} A_l\right) B_2 & \dots & B_{N-1} \end{bmatrix}$ with $\mathbf{M} \in \mathbb{R}^{n \times Nm}$, yields $x_N - \left(\prod_{l=0}^{N-1} A_l\right) x_0 = \mathbf{M} \begin{bmatrix} u_0 & u_1 & \cdots & u_{N-1} \end{bmatrix}^T$.

The linear system is assumed to be controllable, i.e., it is possible to find a control sequence that can reaches any target state from any given initial state of the system in a finite time. Consequently, M is of full rank. The Moore-Penrose solution is of the form

$$u^* = \begin{bmatrix} u_0^* & u_1^* & \cdots & u_{N-1}^* \end{bmatrix}^T = \mathbf{M}^T(\mathbf{M}\mathbf{M}^T)^{-1} \left(x_N - \Big(\prod_{l=0}^{N-1} A_l\Big) x_0\right). \tag{4.50}$$

Similarly, for the subsystem as defined in $P_{st}^{(l)}$,

$$\begin{aligned}\tilde{x}_{j+1}^{s} &= A_j^{ss}\tilde{x}_j^{s} + B_j^{s}u_j + w_j^{s} \\ &= \left(\prod_{l=0}^{j} A_l ss\right)\tilde{x}_0^{s} + \sum_{p=0}^{j-1}\left\{\left(\prod_{l=p+1}^{j} A_l^{ss}\right)\left(B_p^{s}u_p + w_p^{s}\right)\right\} + B_j^{s}u_j,\end{aligned}$$

and in matrix form it is

$$\begin{aligned}&\tilde{x}_N^{s} - \left(\prod_{l=0}^{N-1} A_l^{ss}\right)\tilde{x}_0^{s} \\ &= \begin{bmatrix} \left(\prod_{l=1}^{N-1} A_l^{ss}\right)B_0^{s} & \left(\prod_{l=2}^{N-1} A_l^{ss}\right)B_1^{s} & \left(\prod_{l=3}^{N-1} A_l^{ss}\right)B_2^{s} & \cdots & B_{N-1}^{s} \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_j \\ \vdots \\ u_{N-1} \end{bmatrix} \\ &+ \begin{bmatrix} \left(\prod_{l=1}^{N-1} A_l^{ss}\right) & \left(\prod_{l=2}^{N-1} A_l^{ss}\right) & \left(\prod_{l=3}^{N-1} A_l^{ss}\right) & \cdots & I \end{bmatrix} \begin{bmatrix} w_0^{s} \\ w_1^{s} \\ \vdots \\ w_j^{s} \\ \vdots \\ w_{N-1}^{s} \end{bmatrix},\end{aligned}$$

where w_j^s for $j = 0, 1, \cdots, N-1$ are white Gaussian noise, and all the states will still remain bounded during the system's behavior. Thus, the subsystem is stabilizable and detectable even though some of the states cannot be controlled or observed. ■

Theorem 3 *As F increases on each iteration, the LQT solution of the discrete stochastic decomposition algorithm asymptotically converges to an approximate optimal solution to the discrete LQT problem (4.31)-(4.32).*

Proof. Let l denote the number of iterations, the optimal solution $u_j^{*(l)}$ can be written in the following linear form [25]

$$u_j^{*(l)} = H_j^{1(l)}E_j^{(l)} + H_j^{2(l)} \tag{4.51}$$

where $H_j^{1(l)}$ is the gain for $E_j^{(l)}$ and $H_j^{2(l)}$ captures the remaining terms. As l increases and F becomes sufficiently large, $\Sigma_j^{(l)}$ in (A.8) and $\Psi_j^{(l)}$ in (A.9) approach constant values [25]. Thus, $H_j^{1(l)}$ and $H_j^{2(l)}$ stabilize and the optimal control takes the following linear form

$$u_j^{*(l)} = H_j^{1(\infty)} E_j^{(l)} + H_j^{2(\infty)} \tag{4.52}$$

where

$$\begin{aligned} H_j^{1(\infty)} &= \lim_{l\to\infty} H_j^{1(l)} \\ H_j^{2(\infty)} &= \lim_{l\to\infty} H_j^{2(l)}. \end{aligned}$$

Note that the initial conditions for $E_j^{(l)}$ is $-b$ for all l. Thus, from (4.52), it can be concluded that the solution of the discrete stochastic decomposition algorithm $u_j^{*(l)}$ for $j = 0, 1, \ldots, N-1$ converges to the approximate optimal solution to the discrete LQT problem (4.31)-(4.32).

■

4.3 Summary and Conclusion

This chapter presents a stochastic decomposition method to help reduce the computational complexity for approximating solutions to BIP problems, and then generalizes the method to any LQT with large state space.

The complexity of solving the full-state space LQT problem (3.22)-(3.24) is $O(nm^3)$, which motivates the reduction in the number of constraints. To that end, a reduction is carried out and a Kalman filter is designed to account for the error introduced by the reduction. The final complexity of the resulting procedure (considering reduction and Kalman filtering) is still $O(nm^3)$ for each iteration, but the coefficient is reduced.

Chapter 5

A STOCHASTIC OPTIMAL CONTROL SOLVER FOR APPROXIMATING BIP SOLUTIONS

In Chapter 3, a discrete LQT-based formulation with full-state space is presented and shows much promise to approximately solve large-scale BIP problems efficiently. However, the approximate solutions may be infeasible and/or "superoptimal". Motivated by infeasible-path interior point methods for linear programming, also known as primal-dual interior point methods [44], in this chapter, a new algorithm to balance the feasibility problem with optimizing the objective function is developed. The feasibility pump idea has been adapted to the form of discrete optimal control to further improve the feasibility of the approximate solutions.

5.1 Feasibility and Optimality Approximation LQGT formulations

To approximately solve the original BIP problem (3.1)-(3.3), construct and solve two optimization problems: the full size optimization problem, named the optimality LQT problem, and the constraint satisfaction problem, named the feasibility LQT problem.

The optimality LQT problem, P^o, is

$$\min_{\substack{u_j^o \\ j=0,\ldots,n-1}} \quad \frac{1}{2}\sum_{j=0}^{n-1}\left((x_j^o)^T Q^o(x_j^o) + u_j^o(u_j^o-1)R_j^o\right) + \frac{1}{2}(x_n^o)^T F^o(x_n^o) \tag{5.1}$$

$$\text{s.t.} \quad x_{j+1}^o = x_j^o + a_j^o u_j^o \qquad j = 0,1,\ldots,n-1 \tag{5.2}$$

$$x_0^o = -b^o \tag{5.3}$$

which is constructed from the partial summing formulation, and is equivalent to the full-state space problem (3.22)-(3.24). The state variable is $x_j^o \in \mathbb{R}^{m+1}$ for $j = 0,1,\ldots,n$, and the index j is associated with the partial sum up to j of the m constraints and one objective function. The initial value x_0^o is given by $x_0^o = -b^o = -\left[0,\tilde{b}_1,\ldots,\tilde{b}_m\right]^T$. The control variable $u_j^o \in \mathbb{R}$ is a relaxed form of the binary variable u_j for $j = 0,1,\ldots,n-1$. In

the quadratic criterion (5.1), the parameter R_j^o is a Lagrangian multiplier associated with the constraint $u_j^o(u_j^o - 1) = 0$, and it is a user-specified positive scalar. The cost parameter matrices $Q^o \in \mathbb{R}^{(m+1)\times(m+1)}$ and $F^o \in \mathbb{R}^{(m+1)\times(m+1)}$ are positive semi-definite and user-specified, which are used to penalize the unsatisfied constraints and aide in minimizing the original objective function. In the dynamics equation (5.2), the $(m+1) \times 1$ vector $a_j^o = [\tilde{c}_j, \tilde{a}_{1j}, \ldots, \tilde{a}_{mj}]^T$ for $j = 0, \ldots, n-1$, represents the coefficients $\tilde{a}_{ij}$ in the m constraints and $\tilde{c}_j$ in the single objective function.

The feasibility LQT problem, P^f, is

$$\min_{\substack{u_j^f \\ j=0,\ldots,n-1}} \quad \frac{1}{2}\sum_{j=0}^{n-1}\left((x_j^f)^T Q^f (x_j^f) + u_j^f(u_j^f - 1)R_j^f\right) + \frac{1}{2}(x_n^f)^T F^f (x_n^f) \tag{5.4}$$

$$\text{s.t.} \quad x_{j+1}^f = x_j^f + a_j^f u_j^f \qquad j = 0, 1, \ldots, n-1 \tag{5.5}$$

$$x_0^f = -b^f \tag{5.6}$$

which is similar to the optimality LQT problem P^o but with a different dimension of the state variable. The state variable, $x_j^f \in \mathbb{R}^m$ includes the partial summing variables corresponding to the m constraints but not the original objective function. The initial value x_0^f is given by $x_0^f = -b^f = -\left[\tilde{b}_1, \ldots, \tilde{b}_m\right]^T$. The control variable $u_j^f \in \mathbb{R}$ is a relaxed form of the binary variable u_j for $j = 0, 1, \ldots, n-1$, as u_j^o. In the quadratic criterion (5.4), the cost parameter matrices $Q^f \in \mathbb{R}^{m\times m}$ and $F^o \in \mathbb{R}^{m\times m}$ are positive semi-definite and user-specified, which are used to penalize the unsatisfied constraints in the original BIP. The parameter R_j^f is a Lagrangian multiplier associated with the constraint $u_j^f(u_j^f - 1) = 0$, and it is a user-specified positive scalar. In the dynamics equation (5.5), the $m \times 1$ vector $a_j^f = [\tilde{a}_{1j}, \ldots, \tilde{a}_{mj}]^T$ for $j = 0, \ldots, n-1$.

The key idea of the proposed algorithm is to balance the feasibility problem with optimizing the objective function. To implement the algorithm, define two coupled linear quadratic Gaussian tracking (LQGT) problems, and iteratively solve them in sequence to improve the feasibility and optimality measures with respect to the original binary integer problem. The two problems are coupled through their observations that connect the two LQGT problems.

For the feasibility problem, let l be the iteration index and add a zero-mean Gaussian

noise $w_j^{f,l}$ in the dynamics equation to account for the error introduced in the process. The observation equation is constructed as $y_j^{f,l} = x_j^{f,l} + v_j^{f,l}$, where $v_j^{f,l}$ is a zero-mean Gaussian noise to account for the observation error. The observation $y_j^{f,l}$ is obtained from the state variable $x_j^{o,l-1}$ in the optimality problem from the previous iteration with the first element in $x_j^{o,l-1}$ removed.

The feasibility stochastic approximation LQGT problem, P^{fg}, is

$$\begin{aligned}
\min_{\substack{u_j^{f,l} \\ j=0,\ldots,n-1}} \quad & \mathbb{E}\left\{ \frac{1}{2}\sum_{j=0}^{n-1} \left((x_j^{f,l})^T Q^{f,l}(x_j^{f,l}) + u_j^{f,l}(u_j^{f,l}-1)R_j^{f,l} \right) + \frac{1}{2}(x_n^{f,l})^T F(x_n^{f,l}) \right\} \\
\text{s.t.} \quad & x_{j+1}^{f,l} = \alpha x_j^{f,l} + a_j^{f,l} u_j^{f,l} + w_j^{f,l} && (5.7) \\
& y_j^{f,l} = x_j^{f,l} + v_j^{f,l} \qquad j = 0, 1, \ldots, n-1. && (5.8)
\end{aligned}$$

Similarly, the optimality problem includes a zero-mean Gaussian noise $w_j^{o,l}$ in the dynamics equation to account for the error introduced in the process. The observation equation is constructed as $y_j^{o,l} = x_j^{o,l} + v_j^{o,l}$, where the $v_j^{o,l}$ is a zero-mean Gaussian noise. The observation $y_j^{o,l}$ is obtained as follows: the first element of $y_j^{o,l}$ is not observable but can be calculated from the BIP's objective function, and the rest of the elements of $y_j^{o,l}$ are obtained from the state variable in the feasibility problem $x_j^{f,l}$.

The optimality stochastic approximation LQGT problem, P^{og}, is

$$\begin{aligned}
\min_{\substack{u_j^{o,l} \\ j=0,\ldots,n-1}} \quad & \mathbb{E}\left\{ \frac{1}{2}\sum_{j=0}^{n-1} \left((x_j^{o,l})^T Q^{o,l}(x_j^{o,l}) + u_j^{o,l}(u_j^{o,l}-1)R_j^{o,l} \right) + \frac{1}{2}(x_n^{o,l})^T F(x_n^{o,l}) \right\} \\
\text{s.t.} \quad & x_{j+1}^{o,l} = \beta x_j^{o,l} + a_j^{o,l} u_j^{o,l} + w_j^{o,l} && (5.9) \\
& y_j^{o,l} = x_0^{o,l} + v_j^{o,l} \qquad j = 0, 1, \ldots, n-1. && (5.10)
\end{aligned}$$

The flowchart of the algorithm is shown in Figure 5.1. First construct two stochastic LQGT systems, then iteratively solve the two constructed LQGT problems using the separation principle [8, 12] and get the approximate solution of the control sequence. The initial values for $u_j^{f,0}$, $x_j^{f,0}$, $u_j^{o,0}$, $x_j^{o,0}$ are calculated by solving P^1 and P^2 with the dynamic programming method as discussed in [6]. After each iteration, round the control variables to be binary values and then check if the stopping criteria have been met: 1. infeasibility measure is less than ϵ and optimality measure is less than ϵ; 2. the iteration index is greater

than or equal to the maximum number of iterations. If either of the two stopping criteria has been met, stop the algorithm; otherwise, let $l = l + 1$ and perform a new iteration.

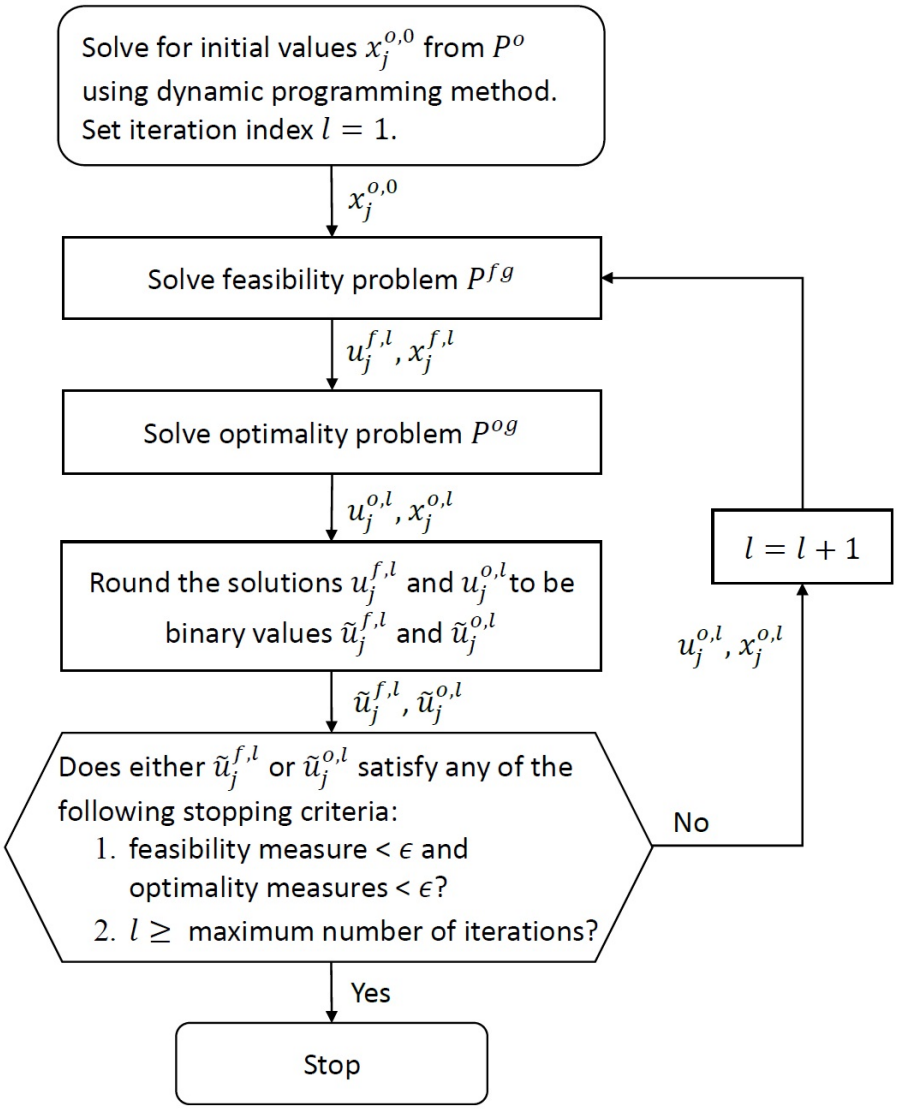

Figure 5.1: Flowchart of the proposed discrete stochastic control algorithm

5.2 Applying Feasibility Pump Idea

The so-called feasibility pump was proposed by Fischetti, Glover, and Lodi [18]. This heuristic turned out to be very successful in finding feasible solutions even for very hard

MIP instances. However, the quality of these solutions in terms of their objective value is sometimes poor. In [1] a slight modification of the feasibility pump was proposed. In contrast to the original version, the modified objective feasibility pump takes the objective function of the MIP into account during the course of the algorithm. Computational results show that the solution quality can indeed be improved by the modified approach without losing the ability to find feasible solutions in a reasonable amount of time.

The following subsection reviews the original version of the feasibility pump as described by Fischetti, Glover, and Lodi [18], and an improved version called the objective feasibility pump as presented by Achterberg, and Berthold [1]. Then, by applying the feasibility pump idea in the form of discrete optimal control, a modification of the proposed discrete stochastic algorithm is introduced in Section 5.2.2.

5.2.1 The Feasibility Pump and Objective Feasibility Pump

A generic MIP can be stated as

$$\min_x \{c^T x : Ax \leq b, x_j \text{ integer } \forall\ j \in I\} \tag{5.11}$$

where A is an $m \times n$ matrix, b is an $m \times 1$ vector, c is an $n \times 1$ vector, and I is the set of indices for which x_j is integer-valued. The feasibility pump heuristic starts by solving the linear programming relaxation of (5.11)

$$\min_x \{c^T x : Ax \leq b\} \tag{5.12}$$

and its solution x_j^* is rounded to the nearest integer point $\tilde{x}_j$ for $j \in I$. For $j \notin I$, $\tilde{x}_j = x_j^*$. Note that $\tilde{x}$ may be infeasible for (5.12). If $\tilde{x}$ is infeasible, the feasibility pump finds a new point that is closest to $\tilde{x}$ in the linear programming polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\}$, by solving the following linear programming problem:

$$\min_x \left\{ \Delta(x, \tilde{x}) = \sum_{j \in I} |x_j - \tilde{x}_j| : Ax \leq b \right\} \tag{5.13}$$

Notice that continuous variables $\tilde{x}_j, j \notin I$, do not play any role in the objective function in (5.13). If the optimal solution, x^*, to (5.13) satisfies $\Delta(x^*, \tilde{x}) = 0$, then x_j^* is an integer

for all $j \in I$, so x^* is a feasible solution for (5.11). If not, the feasibility pump approach rounds x_j^* to an integer point for $j \in I$, and resets $\tilde{x}$ for another iteration. The pair of points $(\tilde{x}, x^*)$ with $\tilde{x}_j$ integer for all $j \in I$ and $x^* \in P$ are iteratively updated at each feasibility pump iteration with the aim of reducing as much as possible the distance $\Delta(x^*, \tilde{x})$. To avoid that the procedure gets stuck at the same sequence of integer and feasible, there is a restart procedure when the previous integer point $\tilde{x}$ is revisited. In a restart, a random perturbation step is performed.

In the original feasibility pump, the objective fucntion of (5.11) is only used at the beginning of the procedure. For the modified objective feasibility pump approach, instead of instantly discarding the objective function of (5.11), it considers a convex combination of the objective functions of (5.11) and (5.13), reducing gradually the influence of the objective term, $c^T x$. The hope is that feasibility pump still converges to a feasible solution but it concentrates the search on the region of high-quality points. The modified objective function is defined as

$$\Delta_\alpha(x, \tilde{x}) := (1-\alpha)\,\Delta(x, \tilde{x}) + \alpha \frac{\sqrt{|\Delta|}}{\|c\|} c^T x, \quad \alpha \in [0,1], \tag{5.14}$$

where $\|.\|$ is the Euclidean norm of a vector and Δ is the objective function vector of (5.13). At each feasibility pump iteration α is geometrically decreased with a fixed factor $\varphi \leq 1$, i.e., $\alpha_{t+1} = \varphi \alpha_t$ and $\alpha_0 \in [0,1]$. Notice that the original feasibility pump algorithm is obtained using $\alpha_0 = 0$. The objective feasibility pump is basically the same as the original feasibility pump algorithm.

5.2.2 Applying Feasibility Pump Idea in the Form of Discrete Optimal Control

To approximately solve the original BIP problem (3.1)-(3.3), the feasibility pump idea has been adapted to the form of discrete optimal control.

In the feasibility LQT problem P^f, if $Q^f = 0$, the problem is equivalent to the following unconstrained problem:

$$\min_{u_j} \quad \frac{1}{2} \sum_{i=1}^{m} \left(b_i - \sum_{j=0}^{n-1} a_{ij} u_j \right)^2 F_i + \sum_{j=0}^{n-1} u_j (u_j - 1) R_j. \tag{5.15}$$

The solution to (5.15), u^F, is rounded to $\widetilde{u}^F$.

In the optimality problem P^o, if $Q^o = 0$, the problem is equivalent to the following unconstrained problem:

$$\min_{u_j} \quad \frac{1}{2}\sum_{i=1}^{m}\left(b_i - \sum_{j=0}^{n-1} a_{ij}u_j\right)^2 F_i + \frac{1}{2}\sum_{j=0}^{n-1}\left(\sum c_j u_j\right)^2 F_0 + \sum_{j=0}^{n-1} u_j\left(u_j - 1\right) R_j. \tag{5.16}$$

The solution to (5.16), u^o, is rounded to $\widetilde{u}^o$.

By applying the feasibility pump idea in the form of discrete optimal control, the following algorithm is developed:

Step 1: solve (5.15), and round its solution $u^{*(l)}$ to binary value $\tilde{u}^{(l)}$.

Step 2: solve a modified version of (5.15) with $\left(u - \tilde{u}^{(l)}\right)^2$, that is

$$\min_{u_j} \quad \frac{1}{2}\sum_{i=1}^{m}\left(b_i - \sum_{j=0}^{n-1} a_{ij}u_j\right)^2 F_i + \sum_{j=0}^{n-1}\left(u_j - \widetilde{u}_j^{(l)}\right)^2 R_j. \tag{5.17}$$

The solution to (5.17), $u^{*(f,l)}$, is rounded to binary value $\tilde{u}^{(f,l)}$.

Step 3: solve a modified version of (5.16) with $\left(u - \tilde{u}^{(f,l)}\right)^2$, that is

$$\min_{u_j} \quad \frac{1}{2}\sum_{i=1}^{m}\left(b_i - \sum_{j=0}^{n-1} a_{ij}u_j\right)^2 F_i + \frac{1}{2}\sum_{j=0}^{n-1}\left(\sum c_j u_j\right)^2 F_0 + \sum_{j=0}^{n-1}\left(u_j - \widetilde{u}_j^{(f,l)}\right)^2 R_j. \tag{5.18}$$

The solution to (5.18), $u^{*(o,l)}$, is rounded to binary value $\tilde{u}^{(l)}$.

Step 4: Check if any of the following stopping criteria is satisfied: the feasibility and optimality measure $\leq \epsilon$, or $\tilde{u}^{(l)}$ converges. If neither is satisfied, set the iteration index l to be $l + 1$, and go back to Step 2.

The flowchart of the proposed algorithm is shown in Figure 5.2.

5.3 Numerical Results

The limits of the two algorithms are explored with six test problems *enigma*, *air*01, *air*03, *air*04, *air*05 and *nw*04 as used in Chapter 3. The dimensions for the test problems and the numerical results are shown in Table 5.1. The results are reported with $Q^f = 0$, $Q^o = 0$, $R_j = 10$, and the diagonal elements of F to $100,000$.

Numerical experiments show that, with the proposed algorithm in Figure 5.1, the feasibility and optimality measures are comparable with the results of the discrete LQT-based

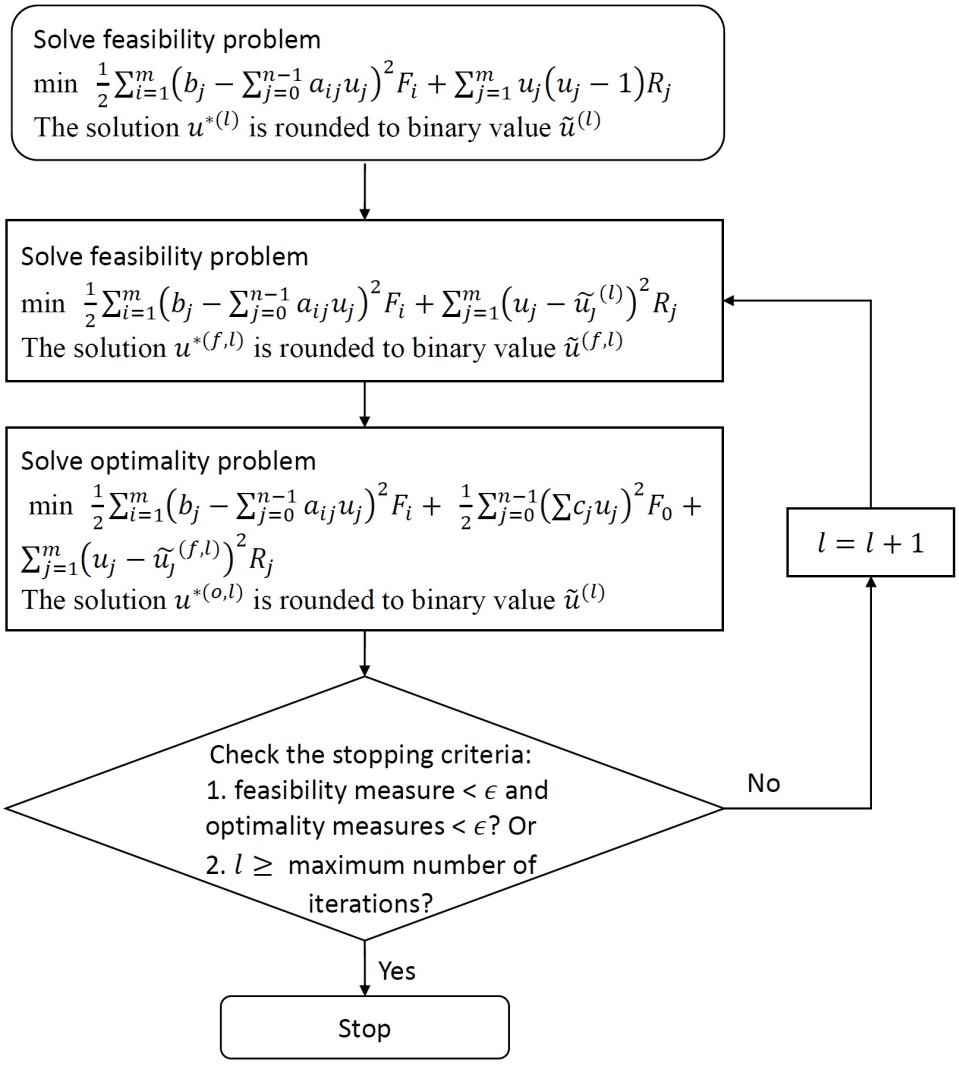

Figure 5.2: Flowchart of the improved algorithm by applying the feasibility pump idea in the form of discrete optimal control

Table 5.1: Test results for algorithms in Figure 5.1 and Figure 5.2

Problem	n	m	Algorithm in Figure 3.1		Algorithm in Figure 5.1		Algorithm in Figure 5.2	
			Feasibility measure	Optimality measure	Feasibility measure	Optimality measure	Feasibility measure	Optimality measure
enigma	100	21	18	0%	20	0%	18	0%
air01	771	23	13	2.55%	22	-26.43%	3	47.39%
air03	10,757	124	138	-11.68%	110	-25.33%	4	66.83%
air04	8,904	823	706	1.43%	811	-27.63%	593	68.75%
air05	7,195	426	322	-0.55%	424	-59.94%	219	30.84%
nw0e4	87,482	36	19	1.36%	36	-9.98%	10	75.44%

algorithm in Figure 3.1, but the computational time is much longer than that with the discrete LQT-based algorithm in Figure 3.1. The reason is that the proposed algorithm in Figure 5.1 is solved iteratively, and within each iteration, two full-state space problems need to be solved. By applying the feasibility pump idea in the form of discrete optimal control, the algorithm in Figure 5.2 further improves the feasibility measure comparing with the algorithm in Figure 5.1. Also, the feasibility measure with algorithm in Figure 5.2 is better than that with the algorithm in Figure 3.1, however, the optimality measure is getting worse.

The comparison of the results in Table 3.3 and the results for algorithm in Figure 5.2 is shown in Figure 5.2. Regarding to the feasibility and optimality measures, the performance of the two algorithms are very close. Regarding to the computational time, the time with the algorithm used in Table 3.3 is much lower because it is based on the full-state space and is solved only once, while the algorithm in Figure 5.2 need to solve two full-state space problem within each iteration.

Table 5.2: Comparison of results in Table 3.3 and results for algorithm in Figure 5.2

Problem	Result from Table 3.3				Algorithm in Figure 5.2			
	Feasibility measure	#violated constraints	#ones in solution	Optimality measure	Feasibility measure	#violated constraints	#ones in solution	Optimality measure
enigma	18	18	1	0%	18	18	1	0%
air01	3	3	12	47.39%	3	3	12	47.39%
air03	3	3	103	68.04%	4	4	102	66.83%
air04	536	471	110	15.91%	593	439	145	68.75%
air05	228	201	60	26.59%	219	194	61	30.84%
nw04	10	10	22	72.18%	10	10	23	75.44%

5.4 Summary and Conclusion

In this chapter, the idea of balancing the feasibility problem with optimizing the objective function has been explored to improve the approximate solution with respect to the feasibility and optimality measures. Numerical results demonstrate the effectiveness of the proposed algorithm by applying the feasibility pump idea in the form of discrete optimal control.

Chapter 6

OTHER APPROXIMATION ALGORITHMS

In this chapter, algorithms that use a different combination of the bang-bang control method, minimum fuel formulation, minimum-variance formulation, and the cross-entropy method have been developed, and their performances for solving large BIP problems have been studied. The basic structure of these algorithms is shown in Figure 6.1.

6.1 Description of the four individual methods

In this section, the four methods, i.e., the bang-bang control method, minimum fuel formulation, minimum-variance formulation, and the cross-entropy method, are briefly introduced.

6.1.1 Bang-Bang Control Method

In optimal control problems, it is sometimes the case that a control is restricted to be between a lower and an upper bound. If the optimal control switches from one extreme to the other, then that control is referred to as a bang-bang control. Bangbang controls frequently arise in minimum time problems. [27]

Here, as suggested in Theorem 1 to solve the discrete LQT problem (3.14)-(3.17), the optimal control, u_j^*, is determined by

$$u_j^* = \begin{cases} 1 & \text{if } \lambda_{j+1}^{*T} a_j < 0, \\ \in [0,1] & \text{if } \lambda_{j+1}^{*T} a_j = 0, \\ 0 & \text{if } \lambda_{j+1}^{*T} a_j > 0. \end{cases} \tag{6.1}$$

The bang-bang control derived in Theorem 1 has two computational difficulties: one is the presence of singular arcs where the bang-bang control is not specified, and the second is the need to solve a two-point boundary-value problem, since the boundary conditions required for solution are the initial state and the final costate. Therefore, the following three methods

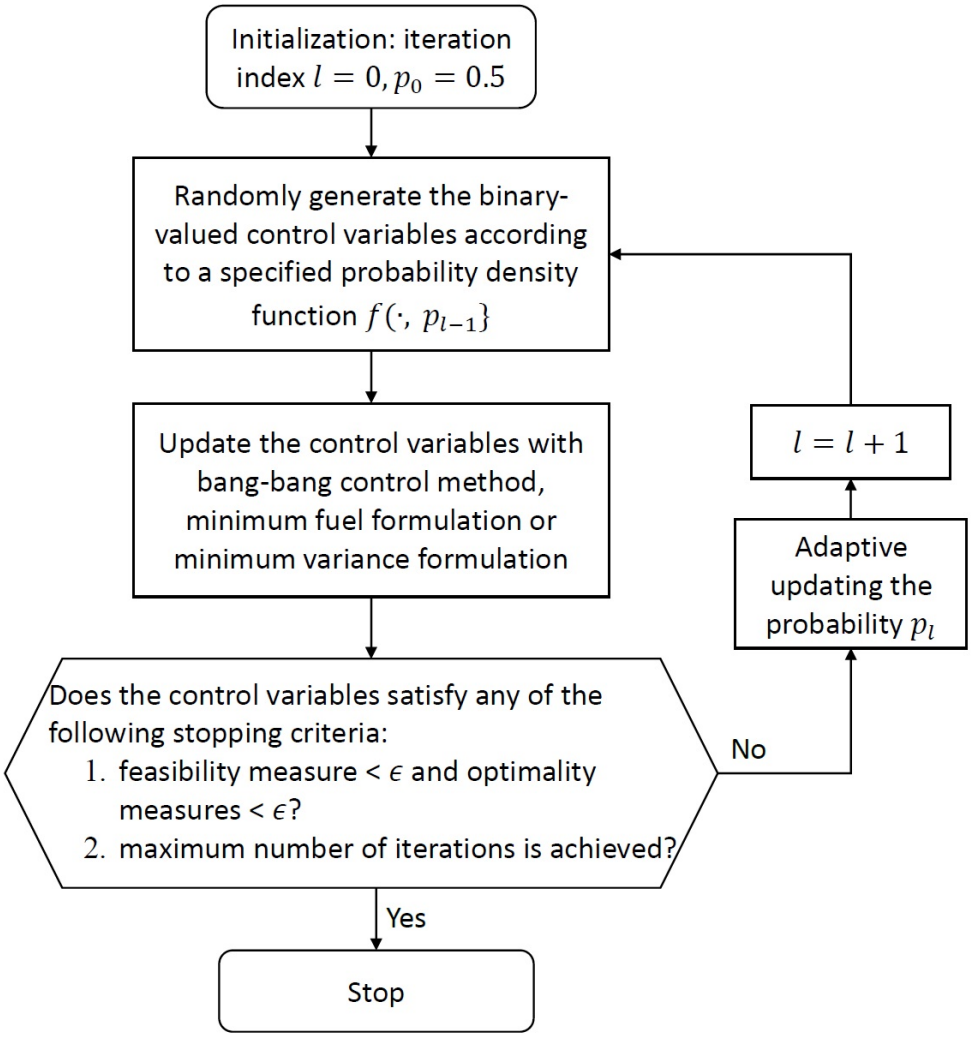

Figure 6.1: Basic structure of the proposed algorithms

are considered to be combined with the bang-bang control method to approximately solve BIP problems.

6.1.2 Minimum Fuel Formulation

The minimum fuel optimal control is important in aerospace applications, where fuel is limited and must be conserved [27]. The minimum fuel formulation for the original BIP problem (3.1)-(3.3) is constructed as follows:

$$\min_{\substack{u_j \\ j=0,\ldots,n-1}} \quad \sum_{j=0}^{n-1} \left(\frac{1}{2} e_j^T Q_j e_j + R_j |v_j| \right) + \frac{1}{2} e_n^T F e_n \tag{6.2}$$

$$s.t. \quad e_{j+1} = e_j + \frac{1}{2} a_j (v_j + 1) \quad j = 0, \ldots, n-1 \tag{6.3}$$

$$-1 \leq v_j \leq 1 \qquad\qquad j = 0, \ldots, n-1 \tag{6.4}$$

$$e_0 = -b \tag{6.5}$$

where $v_j = 2u_j - 1$. It is to find the scalar control v_j to drive the system from a given initial state $e_0 = -b$ to a desired final state e_n as close to zero as possible at a fixed time n using minimum fuel.

6.1.3 Minimum Variance Formulation

The minimum variance problem is constructed as the following quadratic problem in the hope of making $\tilde{e}_j$ and v_j gradually approach zero as the iteration index increases, i.e.,

$$\min_{v_j} \quad I_{j+1} = \mathbb{E} \left\{ \frac{1}{2} \tilde{e}_{j+1}^T Q_{j+1} \tilde{e}_{j+1} + \frac{1}{2} v_j^T R_j v_j \right\} \tag{6.6}$$

for $j = 0, \ldots, n-1$. The state equation is

$$\tilde{e}_{j+1} = \tilde{e}_j + a_j u_j + v_j + w_j \tag{6.7}$$

with the initial state $\tilde{e}_0 = -b$, $u_j \in \{0, 1\}$, and $w_j = |e_j - \tilde{e}_j|$. Here, e_j can be calculated forward from the initial point $e_0 = -b$ to the end point e_n using the dynamics $e_{j+1} = e_j + a_j u_j$. Take the derivative of I_{j+1} over v_j and set it to 0, yielding

$$v_j = -(Q_{j+1} + R_j)^{-1} Q_{j+1} \left(\tilde{e}_j + u_j + w_j \right) \tag{6.8}$$

6.1.4 Cross-Entropy Method

The cross-entropy method was motivated by an adaptive algorithm for estimating probabilities of rare events in complex stochastic networks which involves variance minimization [35]. It was soon realized that a simple cross-entropy modification could be used not only for estimating probabilities of rare events but for solving difficult combinatorial optimization problems as well [36]. Several recent applications demonstrate the power of the cross-entropy method as a generic and practical tool for solving NP-hard problems. [37]

Applying the cross-entropy method to solve the BIP problem involves an iterative procedure where each iteration can be broken down into the following phases:

Step 1: Start with some $\hat{p}_0$, say $\hat{p}_0 = (0.5, 0.5, \ldots, 0.5)$. Let $l := 1$.

Step 2: Draw a sample $U_1, U_2, \ldots, U_N$ of Bernoulli vectors with success probability vector $\hat{p_{l-1}}$ as the initial guesses of the solution to the original BIP problem (3.1)-(3.3), where $U_i = (U_{i1}, \ldots, U_{in})$ for $i = 1, \ldots, N$. Calculate the performances $S(U_i)$, i.e., the feasibility measure as defined in Chapter 3, for all i, and order them from smallest to biggest, $S_{(1)} \leq \ldots \leq S_{(N)}$. Let $\hat{\gamma}_l$ be $(1-\rho)$ sample quantile of the performances: $\hat{\gamma}_l = S_{\lceil (1-\rho)N \rceil}$.

Step 3: Using the same sample to calculate the probability for next iteration, $\hat{p}_l = (\hat{p}_{l,1}, \ldots, \hat{p}_{l,n})$, via

$$\hat{p}_{l,j} = \frac{\sum_{i=1}^{N} I_{S_{(U_i)} \geq \hat{\gamma}_l} I_{\{U_{ij}=1\}}}{\sum_{i=1}^{N} I_{S_{(U_i)} \geq \hat{\gamma}_l}},$$

for $j = 1, \ldots, n$.

Step 4: Check if the stopping criterion is met, for example, gamma does not change for a certain number of subsequent iterations, or probability converges to a binary vector, or max iteration achieved. If yes, then stop; otherwise set $l := l + 1$ and restart a new iteration from Step 2.

In the following sections, algorithms with the combination of above individual methods have been developed and tested. As shown in Figure 6.1, the basic structure of these algorithms is: firstly, randomly generate the binary-valued control variables according to a specified probability density function; secondly, update the control variables with the bang-bang control method, minimum fuel formulation or minimum variance formulation; next, check if the stopping criterion is satisfied, if yes then stop, otherwise, update the probability

and start a new iteration.

6.2 Algorithm with Bang-Bang Control and Minimum Fuel Formulation

Analogous to the shooting method for solving the two-point boundary value problems, the proposed algorithm propagates the state variables forward from the initial boundary values, and propagates the costate variables backward from the terminal boundary values. Then, the control variables are updated using the bang-bang principle. The singularity condition is checked thereafter to decide if the minimum fuel problem needs to be solved to update the control variables. The algorithm is summarized as follows:

Step 1: Randomly generate a sequence of the control variables $u_j \in \{0,\ 1\}$ following Bernoulli distribution with probability $p = 0.5$. Set the iteration index $l = 0$.

Step 2: Propagate the state variables e_j forward from the initial point $e_0 = -b$ to the end point e_n using the dynamics $e_{j+1} = e_j + a_j u_j$.

Step 3: Propagate the costate variables λ_j backward using $\lambda_j = \lambda_{j+1} + Q_j e_j$ from $\lambda_n = F e_n$.

Step 4: Use the bang-bang condition to update all the control variables u_j as follows

$$u_j^* = \begin{cases} 1 \text{ if } \ \lambda_{j+1}^T a_j < 0 \\ 0 \text{ if } \ \lambda_{j+1}^T a_j > 0 \end{cases} \tag{6.9}$$

for $j = 0, 1, \ldots, n-1$. Check the singularity condition. If there is no singular arc, go to Step 6; else, go to the next step.

Step 5: Solve the minimum fuel problem instead to get the updated sequence of u_j and go to Step 6.

Step 6: Check the feasibility and optimality measures, if both of them are smaller than or equal to ϵ, or the maximum number of iterations is achieved, stop; otherwise, set $l := l+1$ and go back to Step 2 to start a new iteration.

The performance of the algorithm is explored with six test problems *enigma*, *air*01, *air*03, *air*04, *air*05 and *nw*04 from MIPLIB as in Chapter 3. Set the diagonal elements of Q_j to be 1, the diagonal elements of F to be 100,000, and the maximum number of iterations to be 20. The dimensions for the test problems and the numerical results are

shown in Table 6.1. And it has been observed that this bang-bang control scheme makes the control variable favor values all zeros or all ones. The reason is that, the test problems all have a small percentage of ones in their optimal solutions as shown in Table 3.1; when a sequence of u_j's are randomly generated as in Step 1, we get more ones than those in the optimal solutions; then in Step 2, the end point $e_n > 0$; and in Step 3, the costate variables $\lambda_j > 0$; next, when using the bang-bang condition in (6.9) to update u_j, we get all zeros; for the next iteration, starting with u_j all zeros, we have $e_n < 0$ and $\lambda_j < 0$, when using the bang-bang condition in (6.9) to update u_j, we get all ones; the loop will repeat until terminated.

Table 6.1: Test results for algorithm with bang-bang control and minimum fuel formulation

Problem	n	m	Feasibility measure	Number of violated constraints	Number of ones in solution	Optimality measure
enigma	100	21	20	20	0	0%
air01	771	23	23	23	0	-32.39%
air03	10,757	124	124	124	0	30.06 %
air04	8,904	823	823	823	0	-124.26 %
air05	7,195	426	426	426	0	-60.04 %
nw04	87,482	36	36	36	0	-13.48 %

6.3 Algorithm with Bang-Bang Control and Minimum Variance Formulation

In this section, the algorithm with a combination of bang-bang control method and the minimum variance formulation is developed. The algorithm is summarized as follows:

Step 1: Generate a random sequence of the control variables $u_j \in \{0,\ 1\}$, for $j = 1, \ldots, n$ following Bernoulli distribution with probability $p = 0.5$. Set the iteration index $l = 0$.

Step 2: Propagate the state variables e_j forward from the initial point $e_0 = -b$ to the end point e_n using the dynamics $e_{j+1} = e_j + a_j u_j$.

Step 3: Solve the minimum variance problem (6.6) to get v_j in (6.8) and update the stochastic state variables $\tilde{e}_j$ in (6.7) forward in time from the initial value $\tilde{e}_0 = -b$.

Step 4: Solve the bang-bang problem to get the costate variables and update the control variables. (i.e., propagate the costate variables λ_j backward using $\lambda_j = \lambda_{j+1} + Q_j\tilde{e}_j$ from $\lambda_n = F\tilde{e}_n$, and use bang-bang principle to update all the control variables u_j.)

Step 5: Check the feasibility and optimality measures, if both of them are smaller than or equal to ϵ, or the maximum number of iterations is achieved, stop; otherwise, set $l := l+1$, update the parameters Q_j, R_j, F, and go back to Step 2 to start a new iteration.

The numerical results for this algorithm are shown in Table 6.2. The results in Table 6.2

Table 6.2: Test results for algorithm with bang-bang control and minimum fuel formulation

Problem	n	m	Feasibility measure	Number of violated constraints	Number of ones in solution	Optimality measure
enigma	100	21	4,494,222	15	46	0%
air01	771	23	23	23	0	-32.39%
air03	10,757	124	124	124	0	30.06 %
air04	8,904	823	823	823	0	-124.26 %
air05	7,195	426	426	426	0	-60.04 %
nw04	87,482	36	36	36	0	-13.48 %

are almost the same as those in Table 6.1, except that the feasibility measure of *enigma* is much worse in Table 6.2. It indicates that the bang-bang condition has a significant impact on the final results, and that there is no significant difference between the minimum fuel formulation and the minimum variance formulation. In Section 6.4, a combination of cross-entropy method and the bang-bang control method is explored to take advantage of the cross-entropy method that provides a simple adaptive procedure for estimating the optimal reference parameters.

6.4 Algorithm with Cross-Entropy and Bang-Bang Control Method

Cross-entropy method is based on a probability density function. It is robust, easy to use and also has asymptotic convergence properties. To take advantage of these good properties, an algorithm with the combination of cross-entropy and bang-bang control method is developed as follows:

Step 1: Start with the initial probability $\hat{p}_0 = (0.5, 0.5, \ldots, 0.5)$. Set iteration index $l = 1$.

Step 2: Generate a random sample u_j for $j = 1, \ldots, n$ following Bernoulli distribution with $\hat{p}_{l-1}$ for N replications, and get the sample vector $U_1, \ldots, U_N$, where $U_i = (u_1, \ldots, u_n)_i$, for $i = 1, \ldots, N$.

Step 3: For each sample vector U_i $(i = 1, \ldots, N)$, propagate the state variables e_j forward from the initial point $e_0 = -b$ to the end point e_n using the dynamics $e_{j+1} = e_j + a_j U_{ij}$, where U_{ij} is the j^{th} element of U_i. Check the value of e_n. If $|e_n| < \epsilon$, go to next step; else, propagate the costate variables λ_j backward using $\lambda_j = \lambda_{j+1} + Q_j e_j$ from $\lambda_n = F e_n$, and use bang-bang principle to update all the control variables U_{ij}.

Step 4: Calculate the maximum the absolute differences of feasibility over all constraints for each sample vector U_i, order them from the smallest to the biggest and get $S_1 \leq ... \leq S_N$. Let $\hat{\gamma}_l$ be the $(1-\rho)$ quantile, i.e., $\hat{\gamma}_l = S_{\lceil (1-\rho)N \rceil}$

Step 5: Update the probability $\hat{p}_l = (\hat{p}_{l,1}, ..., \hat{p}_{l,n})$ via

$$\hat{p}_{l,j} = \frac{\sum_{i=1}^{N} I_{S_{(U_i)} \geq \hat{\gamma}_l} I_{\{U_{ij}=1\}}}{\sum_{i=1}^{N} I_{S_{(U_i)} \geq \hat{\gamma}_l}}$$

Step 6: Check if $\hat{p}_t$ has converged to a binary vector or the maximum iteration has been achieved. If it does, then stop; otherwise, set l to be $l + 1$ and go back to Step 2 to start a new iteration until the maximum number of iterations is achieved.

The numerical test is implemented with $N = 1000$, $\rho = 0.8$, maximum number of iterations to be 20. The dimensions for the test problems and the numerical results are shown in Table 6.3.

Comparing with the results in Table 6.1 and Table 6.2, the performances regarding the feasibility and optimality measures for $air01$, $air03$ and $air05$ have been improved in Table

Table 6.3: Test results for algorithm with cross-entropy and bang-bang control method

Problem	n	m	Feasibility measure	Number of violated constraints	Number of ones in solution	Optimality measure
enigma	100	21	20	20	0	0%
air01	771	23	1	10	1	60.51%
air03	10,757	124	99	38	80	17.21 %
air04	8,904	823	823	823	0	-124.26 %
air05	7,195	426	424	424	1	-59.94 %
nw04	87,482	36	36	36	0	-13.48 %

6.3. And there are no improvements for the other three test problems. Here the performance function $S(U_i)$ used in the cross-entropy method only considers the absolute differences of feasibility over all constraints, not the objective function values. To further improve this algorithm, the performance function $S(U_i)$ need to be carefully constructed.

6.5 Summary and Conclusion

In this chapter, the combinations of the bang-bang control method with minimum fuel formulation, minimum variance formulation, and the cross-entropy method have been studied and implemented. Numerical results show the relatively effectiveness of these algorithms and a potential to be improved in the future.

Chapter 7

SUMMARY AND FUTURE RESEARCH

7.1 Summary

This thesis makes a connection between the BIP problem and the traditional optimal control problem, which gives a different route to solve a BIP problem by transforming it to a classic optimal control problem in a discrete time setting.

Chapter 3 proposes a discrete LQT-based approach to approximate a particular class of BIP problems. More specifically, it defines a partial summing formulation of the original BIP problem, constructs a discrete LQT problem to minimize the sum of infeasibility and another term equal to the objective value of the original BIP problem. The discrete time approach introduces less error than the continuous time approach, and is solved only once for the full-state space to reduce the computational time. Analogous to a classical bang-bang control in continuous time, Theorem 1 proves the existence of an optimal binary solution to the discrete LQT formulation. The numerical results show that the discrete LQT-based approximating method can provide advantages in reducing computation while generating a good approximate solution to BIP problem.

The discrete LQT-based approximation procedure for solving large-scale BIP problems shows much promise because the computational complexity is linear in n (the number of variables) and polynomial in m (the number of constraints), specifically on the order of $O(nm^3)$. However, if the number of constraints is very large, the full-state space may require too much computation, so a stochastic decomposition method, which is an extension of the constraint reduction technique, is developed in Chapter 4. The experience with decomposing the LQT problem for approximating BIP solutions can be generalized to any LQT problem with large state space, and is also presented in Chapter 4. The final complexity of the stochastic decomposition method with Kalman filtering is still on the order of $O(nm^3)$, but the coefficient is reduced.

It is known that BIP problem is NP-hard to solve, and even finding a feasible solution for a generic BIP problem is considered NP-complete. The feasibility pump heuristic turned out to be very successful in finding feasible solutions even for very hard MIP instances. By applying the feasibility pump idea in the form of discrete optimal control, a new algorithm is developed in Chapter 5 to balance the feasibility problem with optimizing the objective function. Numerical experiments show encouraging results with the proposed algorithm to improve the feasibility measures.

In Chapter 6, different combinations of the bang-bang control method with minimum fuel formulation, minimum variance formulation, and the cross-entropy method are studied. Based on the numerical results, for the two algorithms developed in Section 6.2 and Section 6.3, the bang-bang control scheme has a significant impact on the final results, and makes the control variable favor values near zero. For the algorithm developed in Section 6.4, the numerical results show some improvements comparing to the two algorithms developed earlier in Section 6.2 and Section 6.3. All in all, the algorithms developed in this chapter are relatively effective and provide experiences that may lead to further improvements.

7.2 Future Research

The following issues can be explored as possible directions for further research:

- The BIP problem to be solved in this paper is a particular type of general binary integer linear programming (ILP) problems. Specifically, since the constraints are all equalities and there are no continuous decision variables, Problem (3.1)-(3.3) is fairly restricted compared to general ILP problems where the constraints are <= or >=. How to handle problems with inequalities and the generalization for the proposed methods in the thesis to make them useful for general ILP problems would be interesting topics to be explored in the future research.

- The thesis describes a rounding algorithm to recover the binary solution to the original BIP problem from the solution to the optimal control problem. The rounding is simple and intuitive, but it can not guarantee the feasibility of the solution. In the future

research, other quantization methods or further steps should be taken to maintain feasibility.

- To verify and improve the effectiveness of the stochastic decomposition approach for general large LQT problems, more numerical tests need to be done in the future research.

- The computational performances of the approximation algorithms studied in Chapter 6 are not very promising. In the future research, we can try integrating some existing widely-used BIP algorithms in the form of optimal control to approximating BIP solutions.

BIBLIOGRAPHY

[1] Achterberg T. and Berthold T.: *Improving the feasibility pum*p, Discrete Optim., 4 (2007), pp. 77-86.

[2] Ali., M. M.; Khompatraporn, C.; Zabinsky, Z. B. A Numerical Evaluation of Several Stochastic Algorithms on Selected Continuous Global Optimization Test Problems. *Journal of Global Optimization*, Vol. 31 No. 4: 635-672 (2005)

[3] Aoki, M.: *Optimization of stochastic systems: topics in discrete-time systems*, Academic Press: New York (1967)

[4] Atamturk, A., and Savelsbergh, M. W. P.: Integer-programming software systems, *Annals of Operations Research* 140(1): 67-124 (2005)

[5] Barnhart, C.; Johnson, E.L.; Nemhauser, G.L., Savelsbergh; M.W.P.; Vance, P.H. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, *46 (3)*, 316-329 (1998)

[6] Bertsekas, D.P. *Dynamic programming and optimal control*. Volume I, 3rd edition. Athena Scientific (2005)

[7] Bertsimas, D.; Tsitsiklis, J. N. *Introduction to Linear Optimization*. Athena Scientific (1997)

[8] Bryson, A. E. (Jr)., and Ho, Y.: *Applied Optimal Control*. Halsted Press (1975)

[9] Callegari, S.; Bizzarri, F.; Rovatti, R.; Setti G. On the Approximate Solution of a Class of Large Discrete Quadratic Programming Problems by $\Delta\Sigma$ Modulation: The Case of Circulant Quadratic Forms. *IEEE Transactions on Signal Processing*, *58, no. 12*, 6126-6139 (2010)

[10] Callegari, S.; Bizzarri, F. A heuristic solution to the optimisation of flutter control in compression systems (and to some more binary quadratic programming problems) via $\Delta\Sigma$ modulation circuits. *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium*: pp. 1815-1818 (2010)

[11] Caprara, A.; Fischetti, M. Branch-and-cut algorithms. *Annotated Bibliographies in Combinatorial Optimization*. J. Wiley & Sons, Chichester; 45-64 (1997)

[12] Chen, G., Chen, G., and Hsu S. H.: *Linear Stochastic Control Systems*. CRC Press (1995)

[13] Danna, E.; Fenelon M.; Gu, Z.; Wunderling R. Generating Multiple Solutions for Mixed Integer Programming Problems. *Integer Programming and Combinatorial Optimization*. Springer Berlin Heidelberg; pp. 280-294 (2007)

[14] Dantzig, G.; Wolfe, P. Decomposition principle for linear programs. *Operations Research*, 8:101-111 (1960)

[15] Dreyfus, S. E., and Law, A. M.: *The Art and Theory of Dynamic Programming*. Academic Press Inc.: New York (1977)

[16] Fisher, M.L. The Lagrangian relaxation method for solving integer programming problems. *Management Science*, 27:1-18 (1981)

[17] Fischetti M., Glover F., and Lodi, A.: *The feasibility pump*, Math. Program., 104, pp. 91-104 (2005)

[18] Fischetti, M. and Salvagnin D., *Feasibility Pump 2.0*, Math. Program. Comput., 1 (2009), pp. 201-222.

[19] Grötschel, M.; Krumke, S. O.; Rambau, J. *Online Optimization of Large Scale Systems: State of the Art*. Springer Verlag (2001)

[20] Haupt, R. L.; Sue E. H. *Practical genetic algorithms*. Wiley-Interscience (2004)

[21] Hoffman, K. L.: Combinatorial optimization: Current successes and directions for the future. *Journal of Computational and Applied Mathematics* 124: 341-360 (2000)

[22] Jarre, F. Relating max-cut problems and binary linear feasibility problems. Available at www. optimization-online.org (2009)

[23] Jünger, M.; Liebling, T. M.; Naddef, D.; et al.: *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art*. Springer (2009)

[24] Kirkpatrick, S., and Gelatt C. D. Jr., and Vecchi, M. P.: Optimization by simulated annealing, *Science*, 220: 671-680 (1983)

approximation method for stochastic discrete optimization, *SIAM Journal on Optimization*, Vol. 12:479-502 (2002)

[25] Kwakernaak, H., and Sivan R.: *Linear optimal control systems*. John Wiley & Sons, New York (1972)

[26] Lew, A.; Holger M. *Dynamic programming: a computational tool*. Vol. 38. Springer-Verlag New York Incorporated (2007)

[27] Lewis F. L., Vrabie D. L., and Syrmos V. L.: *Optimal control*. John Wiley & Sons (2012)

[28] Linderoth, J. T., and Ralphs, T. K.: Noncommercial software for mixed-integer linear programming. Karlof, J. (ed). *Integer Programming: Theory and Practice*, CRC Press Operations Research Series, pp. 253C303 (2005)

[29] Littleton, J. E.: *An optimal control approach to solving binary integer programs*. Masters Thesis, University of Washington (2005)

[30] Martin, R.K.: *Large Scale Linear and Integer Optimization*. Kluwer, Hingham, MA (1998)

[31] MIPLIB, http://miplib.zib.de/ (2008)

[32] Mitchell, J. E.: Branch and Cut. *Wiley Encyclopedia of Operations Research and Management Science* (2011). DOI: 10.1002/9780470400531.eorms0117

[33] Mitten, L. G.: Branch-and-Bound Methods: General Formulation and Properties, *Operations Research*, Vol. 18, No. 1: 24-34 (1970)

[34] Polak, E.: *Optimization: algorithms and consistent approximations*. New York: Springer (1997)

[35] Rubinstein, R. Y. Optimization of computer simulation models with rare events. *European Journal of Operations Research*, 99: 89-112 (1997)

[36] Rubinstein, R. Y. The simulated entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, 2: 127-190 (1999)

[37] Rubinstein, R. Y., and Kroese, D. P.: *The Cross Entropy Method: A Unified Combinatorial Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*, Springer (2004)

[38] Schrijver, A.: *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, Berlin, Germany (2003)

[39] Shi, L., and Ólafsson, S., Nested partitions method for global optimization, *Operations Research*, 48(3): 390-407 (2000)

[40] Varga R. S. *Matrix Iterative Analysis*. Springer, Berlin, Germany (2000)

[41] von Haartman, K., Kohn, W., and Zabinsky, Z. B.: A meta-control algorithm for generating approximate solutions to binary programming problems. *Nonlinear Analysis: Hybrid Systems*, 2(4): 1232-1244 (2008)

[42] von Haartman, K.: *Trajectory mapping meta-control solver for large scale binary integer programs*. Doctoral Thesis, University of Washington (2009)

[43] Wolsey, L. A.: *Integer Programming*. John Wiley & Sons (1998)

[44] Wright, S. J.: *Primal-dual interior-point methods*. SIAM, Philadelphia (1997)

[45] Zabinsky, Z.B. *Stochastic Adaptive Search for Global Optimization*. Kluwer Academic Publishers, Boston, MA (2003)

[46] Zhang, P., K., Kohn, W., and Zabinsky Z. B.: *A Discrete Meta-Control Procedure for Approximating Solutions to Binary Programs*. Entropy 15, no. 9: 3592-3601 (2013)

Appendix A

DYNAMIC PROGRAMMING APPROACH FOR THE FULL-STATE SPACE LQT PROBLEM (3.22)-(3.24)

In this appendix, we solve for $\hat{u}_j$ in the full-state space LQT problem (3.22)-(3.24) using a dynamic programming approach. The cost-to-go function is written as:

$$V\left(e_j, j\right) = \min_{u_j} \left\{ \ \tfrac{1}{2} e_j^T Q_j e_j + \tfrac{1}{2} u_j (u_j - 1) R_j + V(e_{j+1}, j+1) \ \right\} \tag{A.1}$$

with $V(e_n, n) = \frac{1}{2} e_n^T F e_n$, and equate it to the Riccati form

$$V\left(e_j, j\right) = \frac{1}{2} e_j^T \Sigma_j e_j + e_j^T \Psi_j + \Omega_j, \tag{A.2}$$

where Σ_j represents a symmetric positive-definite $(m+1) \times (m+1)$ matrix, Ψ_j is a positive $(m+1) \times 1$ vector, and Ω_j is a positive scalar.

Combining the equations (A.1), (A.2) and the dynamics in (3.23), yields

$$\begin{aligned} V\left(e_j, j\right) = \min_{u_j} \Bigg\{ & \frac{1}{2} e_j^T Q_j e_j + \frac{1}{2} u_j (u_j - 1) R_j + \frac{1}{2} \Big(e_j + a_j u_j \Big)^T \Sigma_{j+1} \Big(e_j + a_j u_j \Big) \\ & + \Big(e_j + a_j u_j \Big)^T \Psi_{j+1} + \Omega_{j+1} \Bigg\}. \end{aligned} \tag{A.3}$$

In order to minimize this expression, isolate the terms with u_j in them

$$\frac{1}{2} u_j (u_j - 1) R_j + \frac{1}{2} u_j^2 a_j^T \Sigma_{j+1} a_j + u_j a_j^T \Sigma_{j+1} e_j + u_j a_j^T \Psi_{j+1},$$

and take the derivative with respect to u_j and set the value to 0,

$$(u_j - \frac{1}{2}) R_j + a_j^T \Sigma_{j+1} a_j u_j + a_j^T \Sigma_{j+1} e_j + a_j^T \Psi_{j+1} = 0.$$

This yields the solution u_j for the optimal control

$$\hat{u}_j = \frac{\frac{1}{2} R_j - a_j^T \Sigma_{j+1} e_j - a_j^T \Psi_{j+1}}{R_j + a_j^T \Sigma_{j+1} a_j}. \tag{A.4}$$

In order to simplify notation, let

$$S_j = \frac{-a_j^T \Sigma_{j+1}}{R_j + a_j^T \Sigma_{j+1} a_j} \tag{A.5}$$

$$\delta_j = \frac{\frac{1}{2} R_j - a_j^T \Psi_{j+1}}{R_j + a_j^T \Sigma_{j+1} a_j} \tag{A.6}$$

and we can now write

$$\hat{u}_j = S_j e_j + \delta_j. \tag{A.7}$$

Equating the Riccati form $(A.2)$ with the value function in $(A.3)$ evaluated at $\hat{u}_j$ from $(A.7)$, yields

$$\begin{aligned} \frac{1}{2} e_j^T \Sigma_j e_j + e_j^T \Psi_j + \Omega_j = & \frac{1}{2} e_j^T Q_j e_j + \frac{1}{2} (S_j e_j + \delta_j)(S_j e_j + \delta_j - 1) R_j \\ & + \frac{1}{2} \Big(e_j + a_j (S_j e_j + \delta_j) \Big)^T \Sigma_{j+1} \Big(e_j + a_j (S_j e_j + \delta_j) \Big) \\ & + \Big(e_j + a_j (S_j e_j + \delta_j) \Big)^T \Psi_{j+1} + \Omega_{j+1}. \end{aligned}$$

We now solve for Σ_j and Ψ_j by separating the quadratic terms from the linear terms in e_j. Isolating the quadratic terms in e_j, we have

$$\frac{1}{2} e_j^T \Sigma_j e_j = \frac{1}{2} e_j^T Q_j e_j + \frac{1}{2} e_j^T S_j^T R_j S_j e_j + \frac{1}{2} e_j^T \big(I + a_j S_j \big)^T \Sigma_{j+1} \big(I + a_j S_j \big) e_j$$

which yields the Riccati equation corresponding to Σ_j

$$\Sigma_j = Q_j + S_j^T R_j S_j + \big(I + a_j S_j \big)^T \Sigma_{j+1} \big(I + a_j S_j \big). \tag{A.8}$$

Isolating the linear terms in e_j, yields

$$e_j^T \Psi_j = e_j^T S_j^T (\delta_j - \frac{1}{2}) R_j + e_j^T \big(I + a_j S_j \big)^T \Sigma_{j+1} a_j \delta_{j+1} + e_j^T \big(I + a_j S_j \big)^T \Psi_{j+1}$$

and factoring out e_j^T, the tracking equation for Ψ_j is

$$\Psi_j = S_j^T (\delta_j - \frac{1}{2}) R_j + \big(I + a_j S_j \big)^T \Sigma_{j+1} a_j \delta_j + \big(I + \ a_j S_j \big)^T \Psi_{j+1}. \tag{A.9}$$

Therefore, Σ_j and Ψ_j can be found backwards in time by equations (A.8) and (A.9) from initial conditions $\Sigma_n = F, \Psi_n = 0$. Given Σ_j and Ψ_j as in (A.8) and (A.9), $\hat{u}_j$ can be calculated forwards in time from (A.4), (3.23) and (3.24).

To calculate $\bar{u}_j$ in (3.26) for the implementation with quantization, we use the same Σ_j and Ψ_j, but introduce rounding to the nearest integer in (A.4), (3.23) and (3.24) to obtain:

$$\bar{u}_j = \text{int}\left[\frac{\frac{1}{2}R_j - a_j^T\Sigma_{j+1}\bar{e}_j - a_j^T\Psi_{j+1}}{R_j + a_j^T\Sigma_{j+1}a_j}\right]. \tag{A.10}$$

and

$$\bar{e}_{j+1} = \text{int}[\bar{e}_j + a_j\bar{u}_j] \tag{A.11}$$

with $\bar{e}_0 = -\text{int}[b]$.