An ILP approach to Multi Hypothesis Tracking

Erik Liland

Report submitted to

Norwegian University of Science and Technology

Department of Engineering Cybernetics

O.S. Bragstads Plass 2D

Elektroblokk D, Gløshaugen,

7034 Trondheim, Norway

December 2016

Abstract

Autonomous Surface Vessels (ASVs) are, at least in control and navigational perspective, unmanned vessels which can for instance be used to transport cargo and people as well as for surveillance and other tasks. To ensure a safe journey, the route planer needs a real time image of its surroundings in addition to map data to avoid collision and dangerous situations with other vessels. In the maritime environment, this is primarily done by radars mounted on the ship itself, were the challenge is to know which measurement (reflection) belongs together from scan to scan to get a track on each target.

Multiple hypothesis tracking (MHT) has become the preferred method for solving the data association problem in a multi target environment. In a maritime tracking situation for anti collision, MHT demonstrates it reliability for use in safety critical systems since it avoids track coalescence and compensate for missed detections by utilizing multiple scans. This report shows that a Track Oriented MHT (TO-MHT) successfully track targets with detection probability above 70% with less than 5% average track loss in multi target scenarios with clutter.

It also discusses the computational cost of MHT and shows the feasibility for MHT implemented on modest personal computer and potential for paralleling on high end real time systems.

Contents

List of Figures

Nomenclature

AMOS Centre for autonomous marine operations and systems, page i

ASV Autonomous Surface Vessel, page i

AUTOSEA A collaborative research and development project between NTNU AMOS and the Norwegian maritime industry with aim to attain world leading knowledge in design and verification of control systems for ASVs, page i

Clutter Noise in the form of false measurements which is assumed Poison distributed, page i

Dummy measurement a self created measurement at the estimated position (with score 0), page i

Gate an area in which a track expects and approves new measurement to associate with itself, page i

Measurement a point in the measurement space where something is detected, page i

Measurement list a set of measurement which originate from the same scan, page i

Measurement noise Also called observation noise, which is noise that affects the accuracy of the measurement not the existence. White Gaussian with zero mean and covariance R, page i

NTNU Norwegian University of Science and Technology, page i

Scan a procedure which measures the entire area of coverage of the system., page i

Score a measure of the goodness of a measurement-to-track association, page i

System noise Also called process noise, which is noise in the model behaviour. This noise compensates for the uncertainty and non-modelled dynamics of the true system, page i

Target an actual object which the system is trying to track, page i

Track a list of measurement indices, one from each scan, which is believed to originate from the same target, page i

Track hypothesis a new measurement which is inside the gate for an existing track, page i

1 Introduction

1.1 Motivation

This report is the result of TTK4550 Engineering cybernetics specialization report, where the aim is to *let the student specialize in a selected area based on scientific methods, collect supplementary information based on literature search and other sources and combine this with own knowledge into a project report*[1]. The learning outcome of this project is to give the student an extensive knowledge in a current problem, good knowledge in related topics and relevant scientific literature.

1.2 Problem description

The proposed title for this project, and the following master-thesis, was "Multi-target tracking using radar and AIS". The key idea behind this formulation is that radar and AIS have different strengths and weaknesses, and if utilized properly, the strengths of both system can be exploited, while the weaknesses can be reduced. For this project it was decided to focus fully on the gold standard within multi target tracking, namely Multi Hypothesis Tracking (MHT). To ensure the tracking system would be sufficiently understood, correctly implemented and thoroughly tested, the AIS integration was postponed to the master thesis.

The following task was proposed for this project:

- Write a survey on multi-target and multi-sensor tracking methods.
- Implement a multi-target tracking method.
- Describe the method and summarize the findings in a report.

1.3 Goals

The goals for this project is to implement a Track Oriented Multi Hypothesis Tracking algorithm in Python, test this with simulated data and analyze the per-

[1] http://www.ntnu.edu/studies/courses/TTK4550

formance for different conditions and discuss methods for improvement.

1.4 Outline of report

In section 2, different methods for target tracking and data association are presented. Here the focus is to highlight the key difference between the most common tracking methods with focus on the properties these lead to. In section 3, a Track Oriented Multi Hypothesis Tracker (TO-MHT) is thoroughly explained, and in section 4 the integer optimization problem that arises in section 3 is elaborated. The results are presented in section 5 and discussed in section 6.

2 Survey of multi-target tracking methods

The aim of this section is to give the reader a brief overview of tracking as a problem and a feeling for the most popular methods, their assumptions and strong and weak properties as seen from an two dimensional maritime anti collision perspective.

2.1 Tracking

Tracking of an object (target) is the process of estimating its state (i.e. position and velocity) based on discrete measurements from an observation system. An observation system can be a radar, sonar or any other sensor that passively or actively detects objects within an area or volume. Any observation system will be prone to noise, both in form of internal noise and external noise from the environment. This noise will, in varying degree, cause false measurements that the tracking system must take care of. These false measurements are often refereed to as clutter.

2.2 Tracking system

A tracking system can be interpreted as either the complete system from the signal processing level to the finished tracks, or as I define it in this text: *A sys-*

tem that process consecutive measurements from an observation system and collects measurements from the same target into tracks or initiate new tracks. A track is a subset of all the measurements from the observation system that is believed to originate from the same target. The challenge knowing which measurement originating from which (real) target is the core at any tracking system. This association problem is non-trivial even under ideal situations, and the addition of spurious measurements and missed targets only increases the complexity.

There has been developed a large variety of methods to solve this association problem, and most of them have several sub-variants. In the following subsections, some of the most common and popular methods will be presented.

2.3 Nearest Neighbour Filter

The Nearest Neighbour Filter (NNF) is the simplest approach in tracking, where one always selects the closest neighbour as the consecutive measurement in the track, where the distance is the Euclidean distance (1).

$$D(z_k) = \left[(z_k - z_{k-1}) \cdot (z_k - z_{k-1})\right]^{1/2} \tag{1}$$

This approach suffers from being very vulnerable to clutter and dense target scenarios. It can be somewhat improved by estimating an a-priori state through a Kalman Filter and selecting the nearest neighbour to the estimate. This extension is sometimes refereed to as Nearest Neighbour Standard Filter (NNSF) [1] and also differs in that it used the Mahalanobis distance (2) which is a measure of the distance from a measurement to a distribution.

$$D(z_k) = \left[z_k - \hat{z}_k\right]^T S(k+1)^{-1} \left[z_k - \hat{z}_k\right] \tag{2}$$

Under the standard assumption that each target can at maximum generate one measurement, the NNF and NNSF are both single-target methods in the way that they may assign the same measurement to more that one track. They can, however, be expanded to multi-target variants by formulating the problem as a global least squares integer optimization problem, often called Global Nearest Neighbour Filter (GNNF). With this extension, the NNSF is almost becoming a

zero-scan multi hypothesis tracker, in the sense that it seeks to select the optimal combination of mutual exclusive measurement-to-track associations while only looking at the most recent scan. NNF, NNSF and GNNF can be viewed as non-probabilistic models, as they do not assume specific models for noise, clutter, false alarm rate or similar.

2.4 Probabilistic Data Association Filter

Probabilistic Data Association Filter (PDAF) is a *single-target* Bayesian association filter which is based on single scan probabilistic analysis of measurements. The target is assumed initialized and modelled by (8). At each scan, the algorithm calculates the the association probabilities for all the measurement inside a validation gate, with the assumption that at most one of the measurements inside the validation gate is the true target. This leads to the state update equation (3) where the current state is updated with a combined innovation.

$$\begin{aligned} \hat{\mathbf{y}}(k) &\triangleq \sum_{j=1}^{m} \beta_j \tilde{\mathbf{y}}_\mathbf{j} \\ \hat{\mathbf{x}}(k|k) &= \hat{\mathbf{x}}(k|k-1) + \boldsymbol{K}(k)\hat{\mathbf{y}}(k) \end{aligned} \tag{3}$$

where

$$\begin{aligned} \beta_j &= \text{the probability of measurement j to be the correct one} \\ \tilde{y}_j &= y_j - \hat{y}_j \quad \text{the j-th measurement innovation} \\ \boldsymbol{K}(k) &= \text{the Kalman Gain for the k-th time step} \end{aligned}$$

PDAF is computationally modest (approximate 50% more computationally demanding than a Kalman Filter [1] p.163) and have good results in an environment with up to about 5 false measurement in a 4σ validation region [1]. PDAF does not include track initialization and assumes that at most one measurement can originate from an actual target. It also assumes that clutter is uniformly distributed in the measurement space and that the targets history is approximated by a Gaussian with a calculated mean and covariance.

PDAF can be used in multi target scenarios, but only as multiple copies of the single-target filter [2]. PDAF can suffer from track coalescence, which is a phenomenon that merges two tracks into one. This coalescence occurs when two targets have similar paths, and the resulting tracks will be an "average" of the two (actual) tracks. There has been done some work to overcome this coalescence [3].

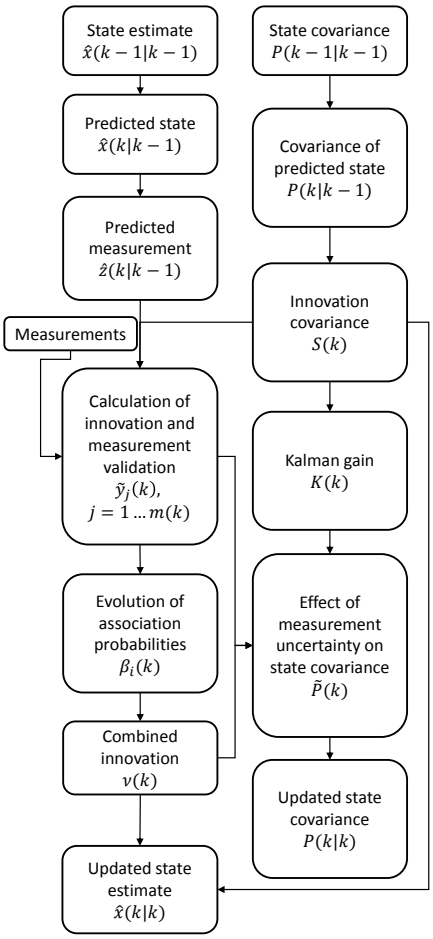


Figure 1: PDAF flowchart

2.5 Joint Probabilistic Data Association Filter

The Joint Probabilistic Data Association Filter (JPDAF) is a *multi-target* extension of the Probabilistic Data Association Filter in which joint posteriori association probabilities are calculated for every target at each scan. Both PDAF and JPDAF use the same weighted sum (3), the key difference is the way the weight β_j is calculated. Whereas PDAF treats all but one measurement inside its validation region as clutter, in JPDAF the targets which interacts (one cluster) are treated as connected and the connected β_js are computed jointly across the cluster set with a given set of active targets inside the cluster. The probability of a measurement j belonging to a target t is [2]

$$\begin{aligned} \beta_j^t &= \sum_{\chi} P\{\chi|Y^k\}\hat{\omega}_{jt}(\chi) \\ \beta_0^t &= 1 - \sum_{j=1}^{m} \beta_j^t \end{aligned} \tag{4}$$

where

$$P\{\chi|Y^k\} = \frac{C^\phi}{c} \prod_{j:\tau_j=1} \frac{exp[-\frac{1}{2}(\tilde{\mathbf{y}}_\mathbf{j}^{\mathbf{t_j}})^T S_{t_j}^{-1}(\tilde{\mathbf{y}}^{\mathbf{t_j}})]}{(2\pi)^{M/2}|S_{t_j}|^{1/2}} \prod_{t:\delta_t=1} P_D^t \prod_{t:\delta_t=0} (1 - P_D^t). \quad (5)$$

Where

$$\beta_j^k = \text{the probability that measurement j belongs to target k}$$

$$\beta_0^k = \text{the probability that no measurement belongs to target k}$$

$$\chi = \bigcap_{j=1}^{m} \chi_{jt_j} \quad \text{All feasible events}$$

$$Y^k = \text{all candidate measurements up to and included time k}$$

$$\hat{\omega}_{jt} = \begin{cases} 1, \text{if } \chi_{jt} \text{ occurs} \\ 0, \text{otherwise} \end{cases}$$

$$m = \text{number of measurements}$$

Since the JPDAF is calculating joint probabilities for all the combinations of measurement associations in the cluster, the computation demand is growing exponentially with the numbers of tracks and measurements in the cluster. A real time implementation of the JPDAF has been developed and patented by QinetiQ [4], and described in [5].

JPDAF also suffers from the same coalescence problem as the PDAF. Seen from a anti collision safety perspective is coalescence not acceptable. An improvement to the JPDAF has been proposed by [3].

Since JPDAF is using all the measurements inside the gate for each track, one measurement can be used to update more than one track if is within more that one gate.

2.6 Multi Hypothesis Tracker

Multiple hypothesis tracking (MHT) is a decision logic which generates and maintains alternative hypotheses when new measurement are received and within the gate. By making several possible hypotheses, the decision in which measurement to choose can be propagated into the future when more information

is available. Each hypothesis is given a score or probability as a measure of the goodness of the measurement, which are accumulated to evaluate the combinations of consecutive measurements.

In contrast to PDA methods which in general will estimate an "average" of two closely spaced tracks as the true track (coalesce), MHT methods split when in doubt. The idea of using multiple hypotheses was first introduced by Singer. et al. [6], but the first complete algorithm was presented by Reid [7], where a hypothesis oriented MHT was developed. Following this, a track oriented MHT was proposed in [8] and the score function for MHT was later deduced and discussed by [9] since no explicit scoring function were given in [8]. MHT is, in the same way as PDAF/JPDAF, developed under the assumption that at most one measurement can originate from each target in each scan, and that a target does not necessary show on every scan (Probability of detection, $P_D < 1$).

Since MHT always selects the best tracks at any time, this can cause apparent inconsistency in the output to the user whenever a new set of measurements are received. If this is unwanted, an alternative representation is to output the N-best tracks and display them to the user in a way that visualises the probability. A third option is to output a weighted average of the tracks along with the covariance of the average.

The MHT approach to tracking and data association was by many for a long time dismissed because of its computationally large cost. The dramatic increase in computational capability from the 1980s to the late 2010s have however lead to a new spring for MHT with an increasing interest for use in tracking system. In 2004 Blackman stated “Multiple hypothesis tracking is generally accepted as the preferred method for solving the data association problem in modern multiple target tracking system” [10]. Already in 2001 did Blackman publish a demonstration that MHT is capable of real-time demands [11].

There are two main approaches to MHT, hypothesis oriented and track oriented.

2.6.1 Hypothesis Oriented MHT

Hypothesis Oriented MHT (HOMHT) or Measurement Oriented MHT (MOMHT) is a tracking approach where direct probabilities of global joint measurement-to-target association hypothesis are calculated. The algorithm initiates tracks and handles missing measurements, it has a recursive nature and allows for clustering for quicker computation.

When a new set of measurements is received, Reids method defines a set of hypotheses, each containing a complete set of associations of the existing tracks and new measurements. By defining the hypotheses in this way, they become compatible in the sense that only one can and need to be selected. When the next set of measurements arrives, each of the current hypotheses are expanded with all measurement-to-track assignments for the new measurements. This way, the hypotheses will keep their compatibility.

When evaluating the alternative hypotheses, each is assigned a probability. This probability takes into account the false-alarm statistics of the measurement system, the expected density of targets and clutter and the accuracy of the target estimates. The probability of each data association hypothesis was developed by Reid in [7].

$$P_i^k = \frac{1}{c} P_D^{N_{DT}} (1 - P_D)^{(N_{TGT} - N_{DT})} \beta_{FT}^{N_{FT}} \beta_{NT}^{N_{NT}} \left[\prod_{m=1}^{N_{DT}} N(\mathbf{Z_m} - \boldsymbol{H}\bar{\mathbf{x}}, \boldsymbol{B}) \right] P_g^{k-1} \tag{6}$$

where

P_i^k = the probability of hypothesis Ω_i^k given measurements up through time k

P_D = the probability of detection

β_{FT} = the density of targets

β_{NT} = the density of previously unknown targets that have been detected

N_{DT} = number of designated target

N_{FT} = the number of false targets

N_{NT} = the number of new targets

N_{TGT} = is the number of targets

$\mathbf{Z_m}$ = the m-th measurement in the current scan

$\boldsymbol{H}$ = the observation matrix

$\boldsymbol{B}$ = the measurement covariance

As with all MHT algorithms, it is crucial to prune the hypothesis tree to avoid an infinite memory and computational cost. In [7], several pruning techniques are used, it most important one is probably to remove all hypotheses with probability below a threshold. A second technique that were proposed is to merge hypotheses with the last N data scan in common.

The initialization of tracks is done through the probabilities of each hypothesis, where a hypothesis would go from a tentative track to a confirmed track when the probability of that hypothesis exceeds i.e. 99%. By treating all new measurement as tentative targets, and calculate their joint probability using information such as density of false targets, density of new targets and probability of detection and thresholding this, the algorithm has an advanced build-in initialization mechanism.

2.6.2 Track Oriented MHT

Track Oriented MHT (TOMHT) is a "bottom-up" approach where the tracks are assumed initialized, and for each scan the track splits whenever there are more than one feasible measurement in the validation region (in addition to the no-

measurement hypothesis). This give rise to a track tree, as in Figure 5, where each initialized track has a root node and multiple branches, were each level represents a scan. A hypothesis is now a set of compatible tracks, with minimum and maximum one track from each track tree, were a hypothesis' score is the sum of its tracks score. Tracks are compatible if they do not share any measurements. The optimal hypothesis is the hypothesis with the highest score.

In the special case were each track tree does not share any measurements (single target), the optimal track is the leaf node with the highest score, which can be found by a traversal of the tree in logarithmic time. However, when a measurement is assigned to two or more targets, the problem of selecting the optimal combination of tracks becomes a multi dimensional assignment problem, since the best hypothesis from each track tree might be mutual exclusive.

New track hypotheses are generated from the filtered estimate from a Kalman Filter, and a score is calculated for instance using (20) from [9]. Both [9] and [10] suggest that modern tracking system could improve performance in certain scenarios by using an Interacting Multiple Model (IMM) approach in stead of a single Kalman Filter. Following the addition of new track hypotheses, the tracks are divided into clusters, in which tracks with common measurements are grouped. The clusters can then be analysed as standalone global problems to find the best possible combination of (possibly mutual exclusive) measurement associations. Linear programming (LP)- and Integer Linear Programming (ILP)-based methods as proposed by [12] can be used to find the best combinations of newly created track hypotheses in accordance to the assumption that a measurement only can be assigned to one target and that one target can maximally create one measurement.

To limit the size of the track hypothesis tree, various pruning schemes has been developed. The maybe single most important pruning is the N-scan pruning, also known as N-scan sliding window. This is a simple technique for limiting the size of the track tree and the following number of leaf nodes. The pruning is done by selecting the node that is N levels higher that the current best leaf node as new root node.

One of the largest drawback with the track oriented MHT approach is the lack of track initialization. This is because TOMHT only considers measurements that are within any (already initialized) targets gate, and automatically discards the unused measurements. The most common approach to add track initialization to TOMHT is to run a separate initialization algorithm in parallel with the tracking loop. This initialization method can be chosen freely, and some popular alternatives are; a dedicated HOMHT running on all or the unused measurements from the main loop, a N-of-M method that can be based on either nearest neighbour or PDA/JPDA or simply a manual initialization by the user. The options for this aiding module are endless, and the specific situation will influence the choice of method.

3 Algorithm walk-through

3.1 Flowchart

Figure 2 shows a flowchart of the track oriented MHT algorithm presented in this section. An important difference between this approach compared to Reid's original measurement oriented MHT [7], is the need for external initialization of targets as mentioned in section 2.6.2.

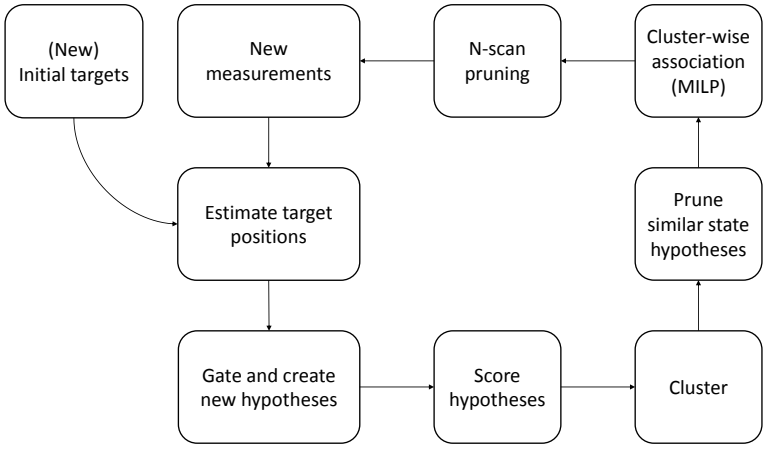


Figure 2: Algorithm flowchart

3.2 State estimation

When a new set of measurement is arriving, it is desirable to "guess" where the target might be before looking for matching measurements. This can be done though a model of the targets dynamics and a state estimator. For the purpose of tracking ships in a local frame (Cartesian plane), we compose a state vector with four states

$$\mathbf{x} = \begin{bmatrix} x & y & \dot{x} & \dot{y} \end{bmatrix}^T \tag{7}$$

where the change of velocity (acceleration) is assumed zero. The latest assumption is compensated with an increased system noise covariance in the model. For a linear system with white and uncorrelated system and measurement noise, the Kalman Filter is an optimal estimator with (8) as time evolution model,

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{\Phi}\mathbf{x}(k) + \mathbf{\Gamma}\mathbf{w} \\ \mathbf{z}(k) &= \boldsymbol{H}\mathbf{x}(k) + \mathbf{v} \end{aligned} \tag{8}$$

where

$$\begin{aligned} \mathbf{\Phi} &= \text{the state transition matrix} \\ \mathbf{\Gamma} &= \text{the disturbance matrix} \\ \mathbf{w} &= \text{the process noise} \\ \boldsymbol{H} &= \text{the measurement matrix} \\ \mathbf{v} &= \text{the observation noise} \end{aligned} \tag{9}$$

. The procedure of predicting the target state at the next time step is according to the "time update" equations of the Kalman Filter (10),

$$\begin{aligned} \bar{\mathbf{x}}(k+1) &= \mathbf{\Phi}\hat{\mathbf{x}}(k) \\ \bar{\boldsymbol{P}}(k+1) &= \mathbf{\Phi}\hat{\boldsymbol{P}}(k)\mathbf{\Phi}^T + \boldsymbol{Q} \end{aligned} \tag{10}$$

where Q is the process noise covariance matrix. The model have the following parameters,

$$\mathbf{\Phi} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{\Gamma} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \boldsymbol{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{Q} = \sigma_v^2 \begin{bmatrix} \frac{T^3}{3} & 0 & \frac{T^2}{2} & 0 \\ 0 & \frac{T^3}{3} & 0 & \frac{T^2}{2} \\ \frac{T^2}{2} & 0 & T & 0 \\ 0 & \frac{T^2}{2} & 0 & T \end{bmatrix} \quad \boldsymbol{R} = \sigma_r^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

where T is the time between the current and the previous measurement, σ_v^2 is the system velocity variance and σ_r^2 is the measurement variance. This model, known as the constant velocity model, is very common due to its simplicity, and is used in among others [7], [13] and [14].

3.3 Gating

To avoid calculating the likelihood for all possible combinations of targets and measurements, some sort of selection criteria is needed when creating new hypotheses. One way of doing this selections is to select all measurements that are within a certain confidence region or gate (11), illustrated in Figure 3.

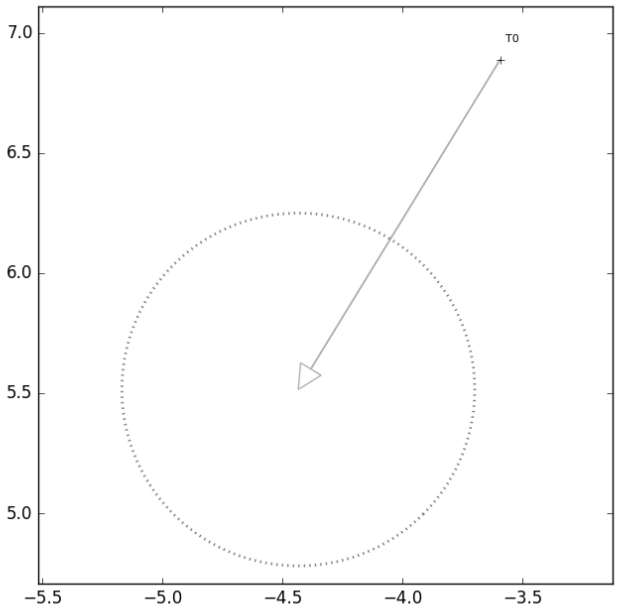


Figure 3: Validation region

$$\begin{aligned} \boldsymbol{B} &= \boldsymbol{H}\bar{\boldsymbol{P}}\boldsymbol{H}^T + \boldsymbol{R} \\ (\mathbf{Z_m} - \boldsymbol{H}\bar{\mathbf{x}})^T \boldsymbol{B}^{-1} (\mathbf{Z_m} - \boldsymbol{H}\bar{\mathbf{x}}) &\leq \eta^2 \end{aligned} \tag{11}$$

This is a very common approach, and is used by most tracking systems at some stage in the processing of new measurements. In (11), η is the chi-square value for a given confidence with degrees of freedom equal to the sensor system. For each measurement inside the region, a filtered state and covariance is calculated using the "measurement update" equations in the Kalman Filter (12), followed by the creation of a new track hypothesis.

$$\begin{aligned} \tilde{\mathbf{y}} &= \mathbf{z} - \boldsymbol{H}\bar{\mathbf{x}} \\ \boldsymbol{S} &= \boldsymbol{H}\bar{\boldsymbol{P}}\boldsymbol{H}^T + \boldsymbol{R} \\ \boldsymbol{K} &= \bar{\boldsymbol{P}}\boldsymbol{H}^T\boldsymbol{S}^{-1} \\ \hat{\mathbf{x}}(k) &= \bar{\mathbf{x}} + \boldsymbol{K}\tilde{\mathbf{y}} \\ \hat{\boldsymbol{P}}(k) &= (\boldsymbol{I} - \boldsymbol{K}\boldsymbol{H})\,\bar{\boldsymbol{P}} \end{aligned} \tag{12}$$

The χ^2 value for selected confidence values are listed in Table 1, where $95\% - 99\%$ is a common interval for gate size.

Confidence	70%	80%	90%	95%	97.5%	99%	99.5%
η^2	2.41	3.22	4.61	5.99	7.38	9.21	10.60

Table 1: χ^2 values for two degrees of freedom

3.4 Scoring

Each track hypothesis are scored according to [9]:

$$\begin{aligned} \mathrm{NLLR}_{t,j}(k) &= \frac{1}{2}\left[\tilde{y}_k^T S_{tj}(k)^{-1}\tilde{y}_k\right] + \ln\frac{\lambda_{ex}|2\pi S_{tj}(k)|^{1/2}}{P_{D_t}(k)} \\ \tilde{y}_k &= z_j(k) - \hat{z}_t(k|k-1) \end{aligned} \tag{13}$$

The cumulative NLLR is then

$$l_t^k \triangleq \sum_{l=0}^{k} NLLR_{t,j(t,l)}(l) \tag{14}$$

3.5 Clustering

Since the global problem of finding the optimal selection of hypotheses is growing exponentially with the number of hypotheses, it is computationally beneficial to split the problem into smaller problem. This can only be done to targets that does not share any measurements. The clustering can be done efficiently through breath-first-search or depth-first-search on a graph made from the hypothesis tree.

3.6 Association

When the targets are divided into independent clusters, each of them can be treated as a global problem where we want to minimize the cost or maximize the score of the selected track hypotheses (leaf nodes), while fulfilling the constraints that each measurement can only be a part of one track and that minimum and maximum one hypothesis can be selected from each target. Since only binary values, selected of not selected, is desired for selection of hypotheses, the problem becomes a integer linear optimization problem (ILP). The solution to this problem is explained in section 4.

3.7 N-scan pruning

To keep the computational cost within reasonable limits, it is necessary to limit the amount of time steps backwards in time that the algorithm computes. This is done by removing all but the active hypothesis at the current root node, and assign the remaining hypothesis as new root node. A similar, and in some scenarios perhaps better, strategy is N-observation pruning proposed by [10]. The idea is to prune N *measurements* back in time instead of N *scans*. This approach will lead to larger and more costly track trees for tracks that have low probability of detection. Therefore, in any practical implementation a N-observation pruning scheme will be followed by a traditional M-scan pruning, where M is larger than N. The combination of these two methods can be seen on as a form of adaptive pruning where tracks with high probability of detection can manage

with fewer history steps than tracks with low probability of detection.

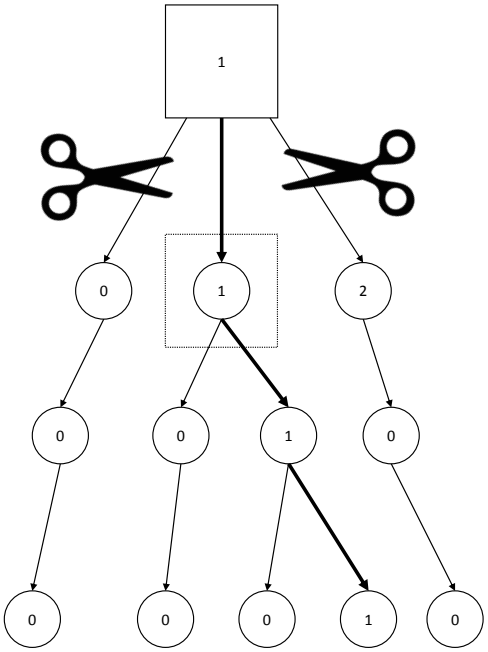


Figure 4: Pruned track hypothesis tree

4 Linear programming

The aim of this section is to elaborate the use of linear programming to solve the data association problem in MHT that arises when there are multiple (possible mutual exclusive) possibilities of measurement arrangements within the existing set of tracks. As with any optimization problem, we need an objective function which tells us how good or bad a given assignment is, and a set of constraints that limits the solution to physical limits and our assumptions.

4.1 Problem formulation

Multiple optimization formulations of the association problem for multi target tracking have been proposed through the history. The first was [15] where a 0-1 ILP was proposed, followed by [12] where an ILP scheme with LP relaxation

and Greedy Rounding Procedure (GRP) as solvers were proposed and [16] were a framework for both association and removal of competing tracks.

All three mentioned publications have essentially equal objective functions, though some use minimize and other use maximize in their formulation. The essence of them all is (15), where $\mathbf{c}$ is a vector of costs (minimize) or scores (maximize) and $\mathbf{x}$ (or $\boldsymbol{\tau}$ as it is also commonly called) is a selection vector, where each row in $\mathbf{c}$ and $\mathbf{x}$ represents one branch in the track hypothesis tree.

$$\min_{x} \quad \mathbf{c}^T\mathbf{x} \tag{15}$$

Further, the constraints that shall ensure that each measurement is not assigned to more than one track are generally formulated as (16) or (17) , where $\boldsymbol{A}$ is a binary matrix whose rows are branches in a track hypothesis tree and $\mathbf{b}$ is a vector with ones.

$$\begin{aligned} &\boldsymbol{A}\mathbf{x} = \mathbf{b} \\ &\mathbf{x} \in \{0,1\} \end{aligned} \tag{16}$$

$$\begin{aligned} &\boldsymbol{A}\mathbf{x} \leq \mathbf{b} \\ &\mathbf{x} \in \{0,1\} \end{aligned} \tag{17}$$

The difference between (16 and (17) originates from the

$$\begin{aligned} \text{maximize} \quad & \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & \boldsymbol{A_1}\mathbf{x} \leq \mathbf{b_1} \\ & \boldsymbol{A_2}\mathbf{x} = \mathbf{b_2} \\ & \mathbf{x} \in \{0,1\} \end{aligned} \tag{18}$$

Where $\boldsymbol{A_1}$ is a $N_1 \times M$ binary matrix with N_1 real measurements and M track hypotheses (all leaf nodes), where $\boldsymbol{A_1}(i,j) = 1$ if hypothesis j are utilizing measurement i, 0 otherwise. The measurements and hypothesis are indexed by the order they are visited by a depth first search (DFS). $\boldsymbol{A_2}$ is a $N_2 \times M$ binary matrix where N_2 is the number of targets in the cluster and $\boldsymbol{A_2}(i,j) = 1$ if hypothesis j belongs to target i. $\mathbf{b_1}$ is a N_1 long vector with ones and $\mathbf{b_2}$ is a N_2 long vector with ones. $\mathbf{c}$ is a N long vector with a measure of the goodness of the track

hypotheses. For example in Figure 5 at time step 2, the A matrices and C vector would be:

$$
\begin{aligned}
\boldsymbol{A_1} &= \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{b_1} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \\
\boldsymbol{A_2} &= \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \mathbf{b_2} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\
\mathbf{c} &= \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & \lambda_4 & \lambda_5 & \lambda_6 & \lambda_7 & \lambda_8 & \lambda_9 \end{bmatrix}^T
\end{aligned}
\tag{19}
$$

4.2 Solvers

There are a lot of of-the-shelf integer linear program (ILP) and mixed integer linear program (MILP) solvers on the marked, both free open source and commercial. Since our problem is formulated on standard form, it can easily be executed on several solvers, and we can compare runtime and performance. In this report, the following solver are tested:

- CBC (Free, COIN-OR)
- CPLEX (Commercial (Free academic), IBM)
- GLPK (Free, GNU)
- Gurobi (Commercial (Free academic), Gurobi)

5 Results

5.1 Testing scheme

The evaluation of the MHT algorithm is two-sided, firstly the algorithm must be able to track under challenging conditions, secondly it must be able to do this

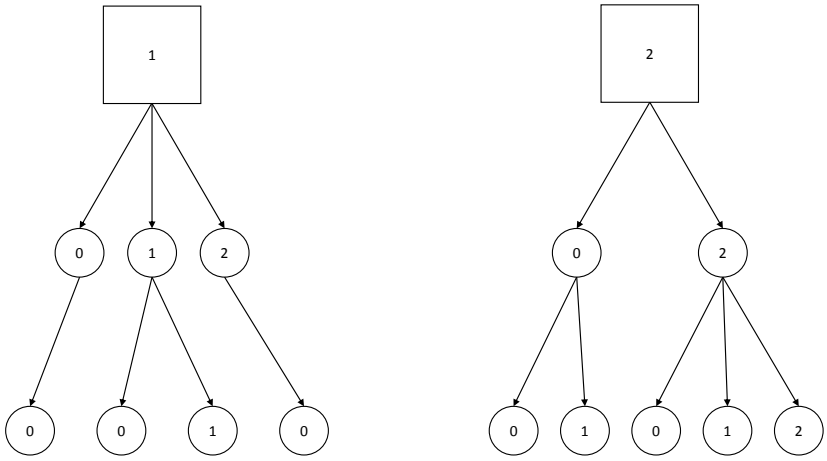


Figure 5: Track hypothesis tree

without having an ever growing computationally cost and run time. The first performance metric is how well the algorithm is estimating the true position to the object it is tracking. This is measured as the Euclidean distance between the estimated and true track:

$$\Delta P = \|\mathbf{p}_{track} - \mathbf{p}_{target}\|_2 \tag{20}$$

where the track is considered correct if $\Delta P \leq \varepsilon_p$ for all t after initial convergence. If a track is deviating more that the threshold and is never within the threshold again, it is considered lost at the time-step it exceeded the threshold. If the track should converge after exceeding the threshold, it is considered restored at the time-step it is returning within the limit. The algorithm is tested on six scenarios:

- Five fully cooperating ships
- Five partially cooperative ships
- Five ships avoiding obstacles with large space

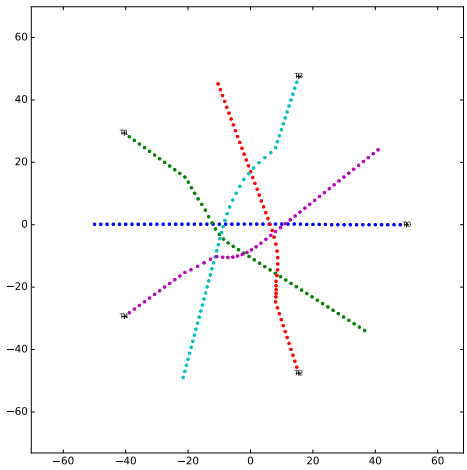
- Five ships avoiding obstacles with little space
- Five almost parallel ships (normal speed radar)
- Five almost parallel ships (high speed radar)

5.2 Simulation data

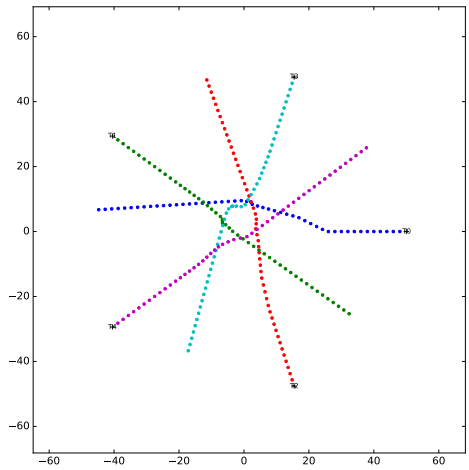
Scenario one through four was generated as a recording of time and position from an Autonomous Surface Vessel (ASV)-simulator with collision avoidance (COLAV), which is a project under work by D. Kwame Minde Kufolaor at NTNU. The ships are configured such that they need to maneuver to avoid collision with each other, which allows for tracking of maneuvering targets in close proximity to each other. These scenarios where sampled at 1 Hz which is a little faster than a normal high speed vessel radar at 0.8 Hz (48 rpm). The fifth scenario was generated as a part of this project, and is composed by linear parallel paths with white Gaussian system noise as maneuvering. This data set where sampled at 0.5 Hz which is a little higher that a normal maritime costal radar at 0.4 Hz (24 rpm) and at 1 Hz to compare the two most common radar update frequencies. Scenario 1 to 4 have a sensor field of view (FOV) of a Table 2 shows the expected number of clutter measurements in each of the two different fields of view.

			$\lambda_\phi \cdot FOV$				
Scenario	Radar range	FOV	1	2	4	6	8
1-4	$65m$	$1.3 \cdot 10^4 m^2$	1.3	2.6	5.2	7.38	9.21
5-6	$155m$	$7.5 \cdot 10^4 m^2$	3.22	4.61	5.99	7.38	9.21

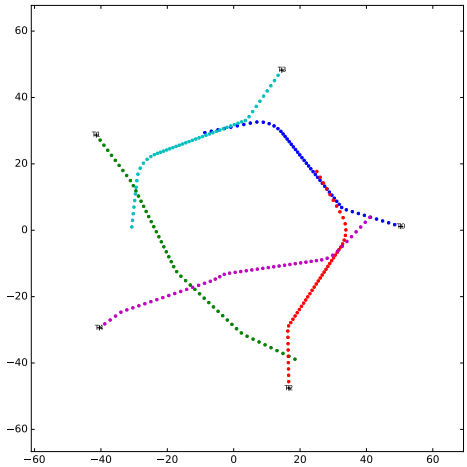
Table 2: Expected number of clutter measurements in scenarios



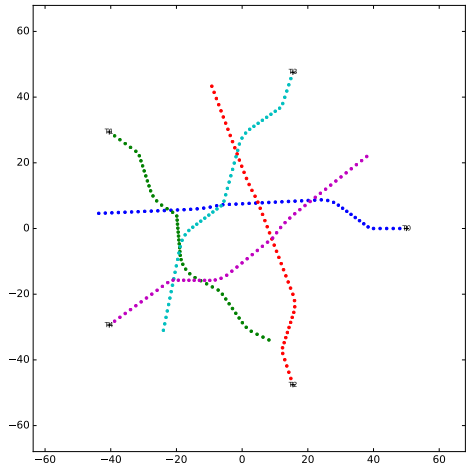
(a) First scenario



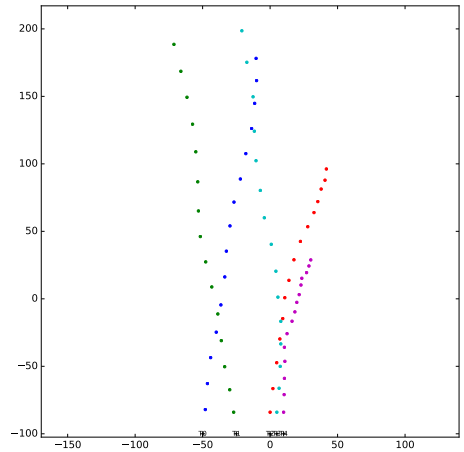
(b) Second scenario



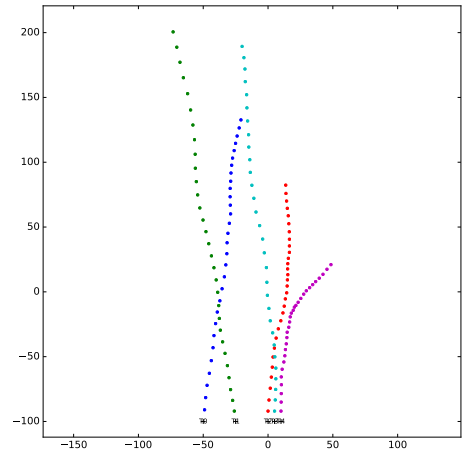
(c) Third scenario



(d) Fourth scenario



(e) Fifth scenario



(f) Sixth scenario

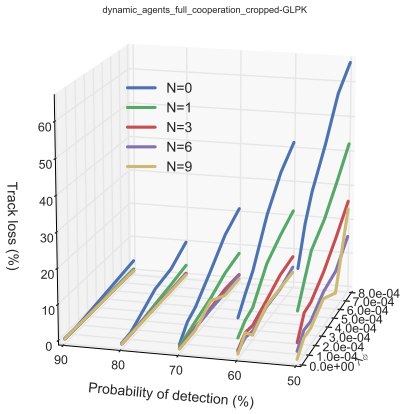
Figure 6: True track for scenario 1-6

5.3 Simulations

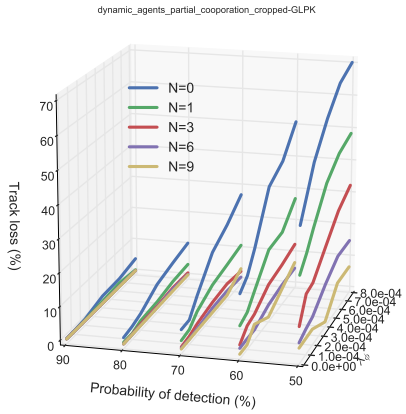
Scenario one through four was simulated with the following variations with all four solvers.

$$\mathbf{P_D} = \begin{bmatrix} 0.5 & 0.6 & 0.7 & 0.8 & 0.9 \end{bmatrix}$$
$$\mathbf{N} = \begin{bmatrix} 0 & 1 & 3 & 6 & 9 \end{bmatrix}$$
$$\boldsymbol{\lambda_\phi} = \begin{bmatrix} 0 & 1 \cdot 10^{-4} & 2 \cdot 10^{-4} & 4 \cdot 10^{-4} & 6 \cdot 10^{-4} & 8 \cdot 10^{-4} \end{bmatrix}$$

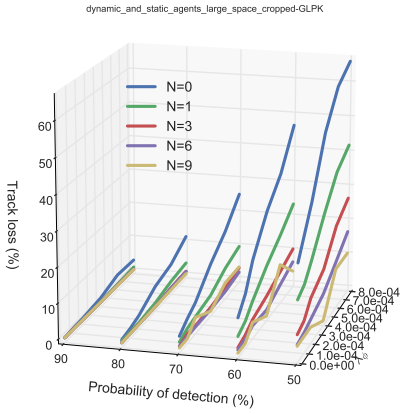
Each variant was simulated 160 times with different seeded random clutter measurements and miss detections. For each simulation, the estimated tracks where compared with the true tacks and categorised in successful and lost tracks, the track loss threshold ϵ_p was 4 meters. Figure 14 - 17 show the averaged track loss percentage for all simulations variants. Since the solvers only differ when it comes to run time and not track performance, the plots from the different solvers are identical and Figure 7 shows the track performance in a more compact manner.



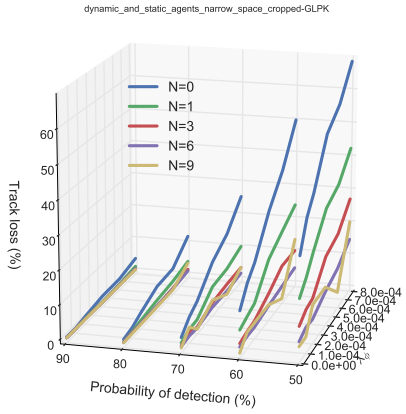
(a) Scenario 1



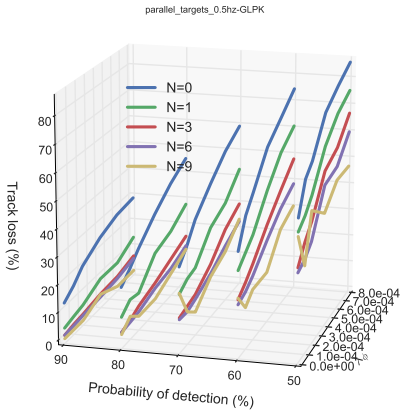
(b) Scenario 2



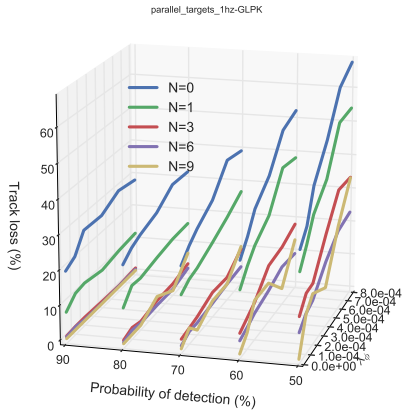
(c) Scenario 3



(d) Scenario 4



(e) Scenario 5



(f) Scenario 6

Figure 7: Simulation results for all scenarios

5.4 Runtime

Figure 8 to 12 displays average the runtime for the entire scenario for the different solvers.

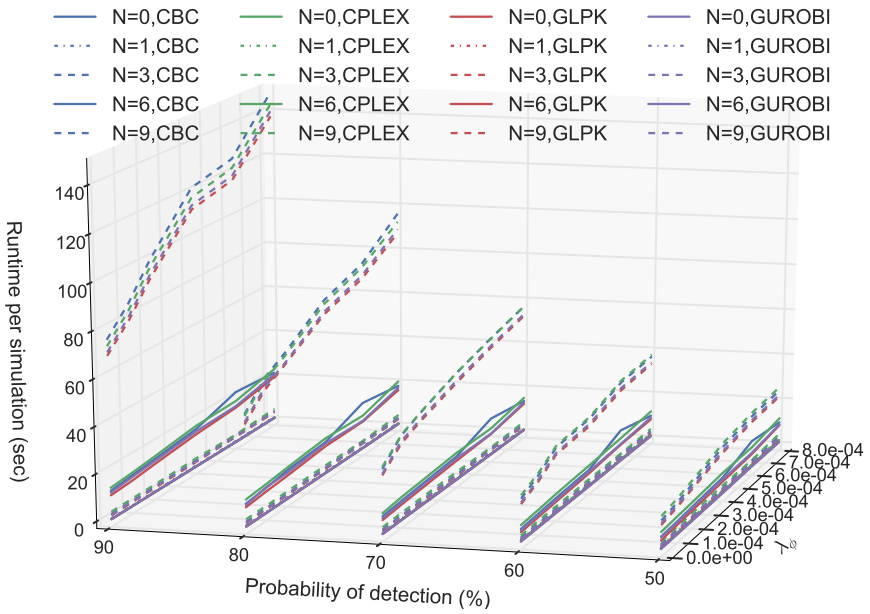


Figure 8: Runtime for scenario 1

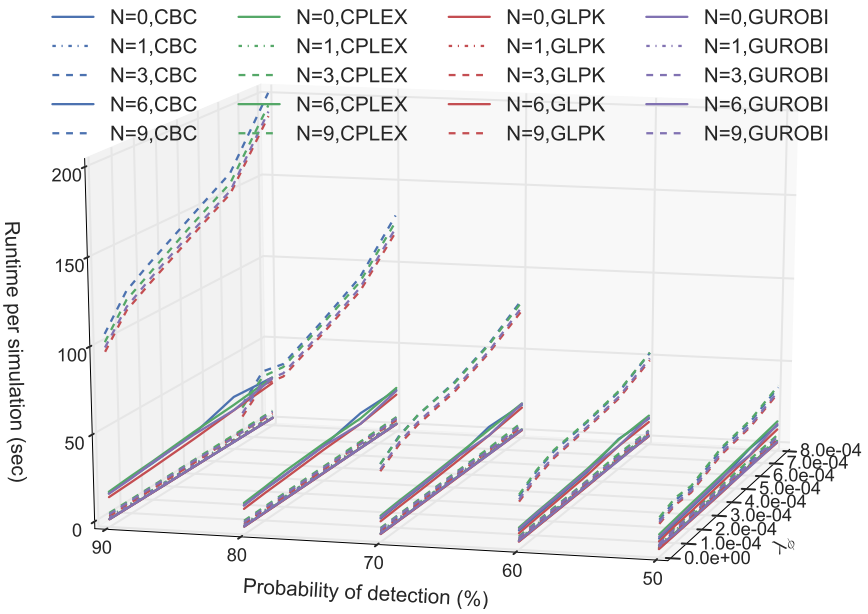


Figure 9: Runtime for scenario 2

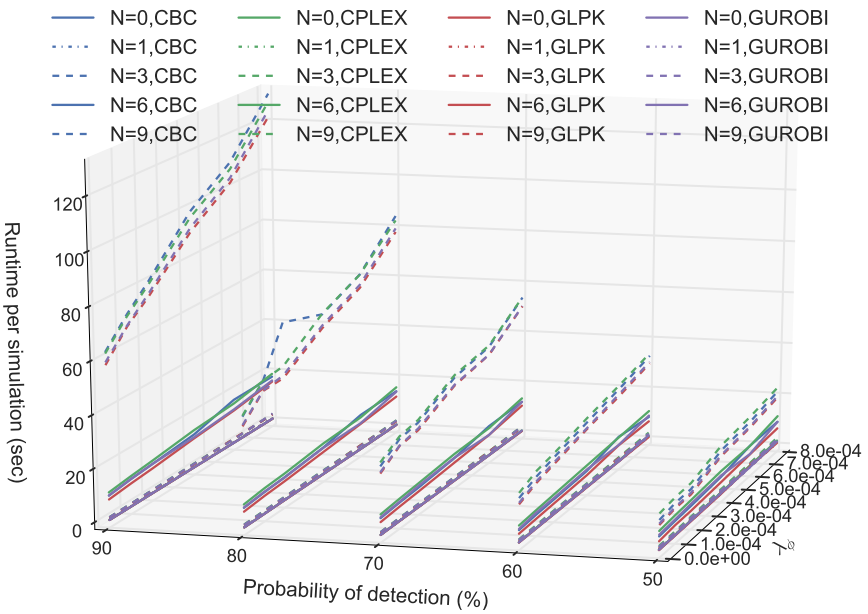


Figure 10: Runtime for scenario 3

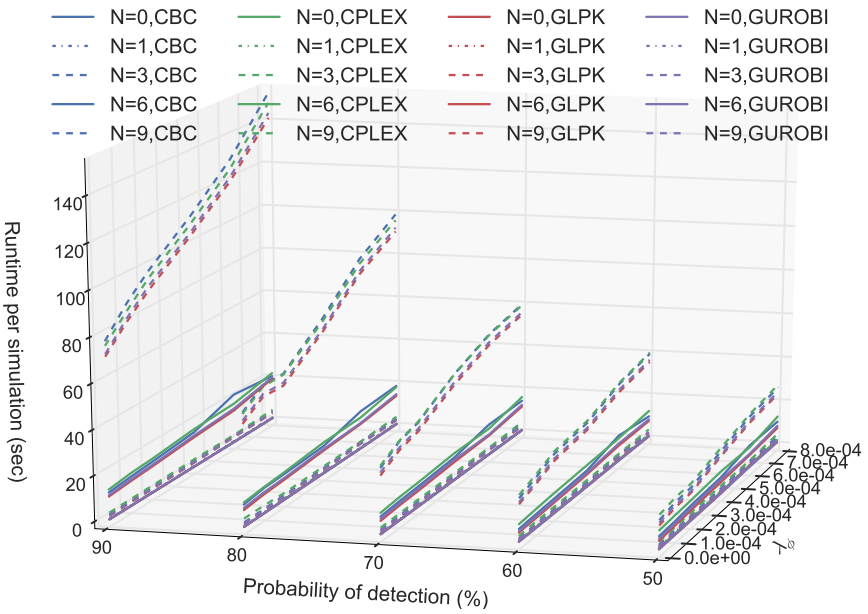


Figure 11: Runtime for scenario 4

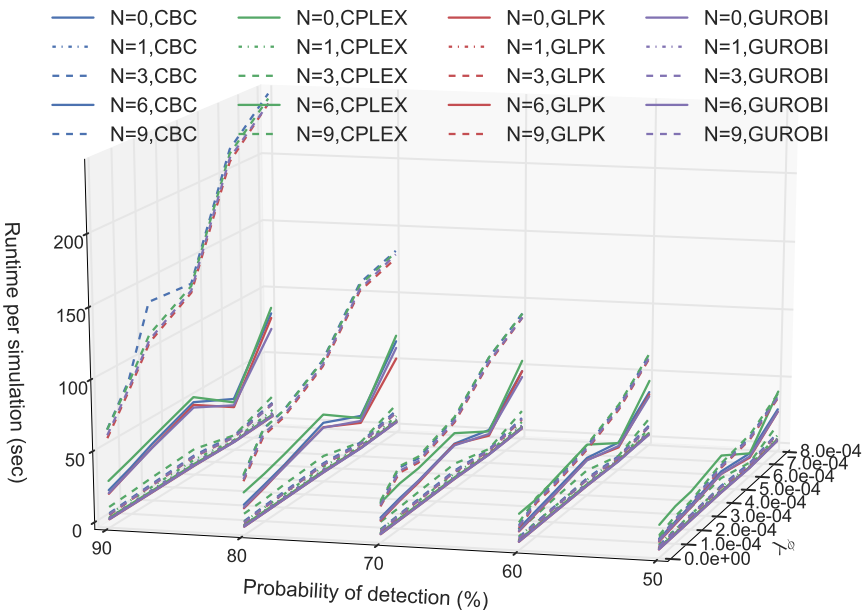


Figure 12: Runtime for scenario 5

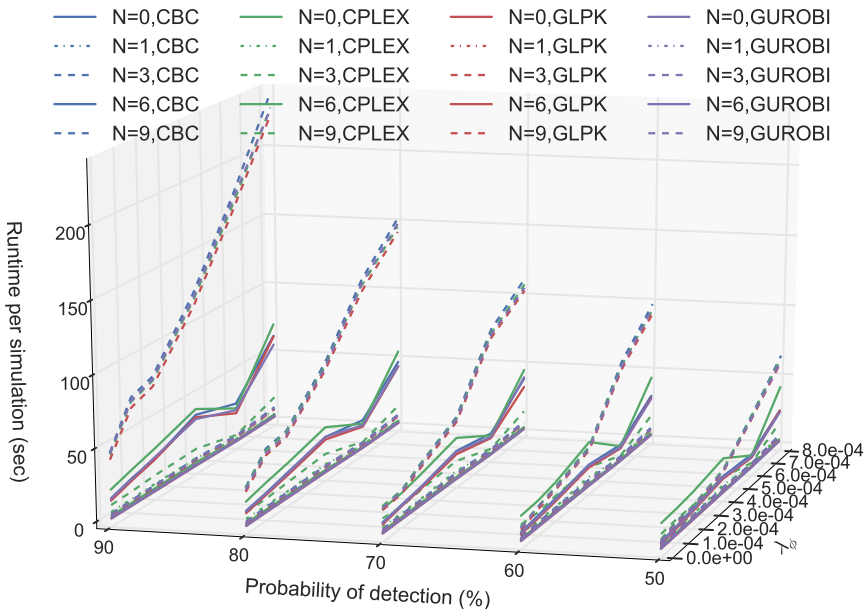


Figure 13: Runtime for scenario 6

5.5 Track loss performance

From the simulation results in Figure 14 to 18 it can be observed that the different solvers perform practically identically when it comes to track performance. From Figure 7, it can be seen that the number of lost tracks is proportional to the clutter level at an inverse proportional rate to the probability of detection P_D. An interesting observation is the return on investment regarding the number of scans to evaluate (N-scan). For instance, with $P_D = 0.7$ the difference between $N = 3$ and $N = 6$ is marginal, at least when the clutter level is within reasonable levels ($\lambda_\phi < 4 \cdot 10^{-4}$). With $P_D = 0.5$ we see that the pay-off is much higher for the extra computational cost with 15% improvement between $N = 3$ and $N = 6$.

5.6 Time performance

From Figure 8 to 13 it can be seen that the solvers are all in the same area, though with a very consistent difference in scenario 1 to 4. In these scenarios the GLKP solver is the fastest with between 5-10 seconds compared with CPLEX which is

the slowest in these scenarios. This trend is most likely caused by the different amount of overhead in the solvers, where IBM's CPLEX which is marked leading in many ways might have way more initialization and setup procedures compared GNU's GLPK. In scenario 5 and 6, a different and more varying trend is visible. Here we see that both GUROBI and GLPK are fastest in different situations, and generally all other solvers than CPLEX perform quite similar.

The run time increases very linear with the amount of clutter, which is expected as most of the operations run in the algorithm are based on tree operations with $O(E + V)$ run time. The structure in a track-tree implies that each vertex has one edge, hence any depth-first-search based operations will have run time $O(2V) = O(V)$. The run time increases near exponentially with the size of N, which is very naturally since the number of hypotheses that must be considered is exponentially larger.

6 Discussion

The field of tracking and data association have been dominated by first military applications and secondly civil aviation control, both of which are areas with hight performance demands and large development contracts. Although much work is published in this topic, few comparisons have been made between actual implementations of different trackers and filters. This is most likely due to the nature of business secrets and military classifications to hide the intricate details that goes into an actual implementation and optimization of any tracking system.

One of the main objective for this work was to examine the feasibility of using off-the-shelf solvers to solve the assignment problem in a MHT system. And does it? The short answer is yes, but with a quite large set of drawbacks. When using off-the-shelf solvers, the assignment problem must be written to a file that the solver can interpreter. This is a major drawback with this approach, but can be partially compensated for with fast solid state storage (SSD) in stead of traditional rotating hard drives. Another major drawback is the need of explicit enumeration of all hypotheses in the ILP formulation, which is costly and in many

situations unnecessary. This could possibly be addressed by column generating optimization methods, which starts with a feasible set of columns and adds more and more columns (constraints) until it reaches the optimal assignment. Intuitively this makes sense, since most of the constraints will never be active, and therefore not needed for finding the optimal solution.

7 Closing remarks

7.1 Conclusion

This report has shown that off-the-shelf ILP solvers, both free and commercial, are capable solving the data association problem in a track oriented multiple hypothesis tracker. It has also shown that the tracking performance does benefit very little from more than $N = 3$ scan history when the probability of detection $P_D > 70\%$, even with severe amounts of clutter.

7.2 Future work

One area where there might be substantial benefits is in more complex pruning and merging of hypotheses. By nature, many hypotheses will have very similar paths and could therefore be candidates for being merged to one. The danger with this is that the assignment problem could render infeasible and crash the algorithm. A novel approach could be to formulate the pruning as an optimization problem. With this approach it is possible to also remove hypotheses with a low score, subject to the problem still being feasible.

I the maritime context, it could also be very beneficial to use the maritime AIS (Automatic Identification System) to aid the tracking. There are different approaches to include this extra set of data, with their pros and cons. One option could be to use the AIS track to weight the measurements from the radar, this way the measurement that are more likely to originate from a target is given a higher score. Another possibility is to do track-to-track fusion / filtering, where the best or N-best tracks from the radar are filtered with the tracks from the AIS system. A similar problem is tackled by Coraluppi in [13].

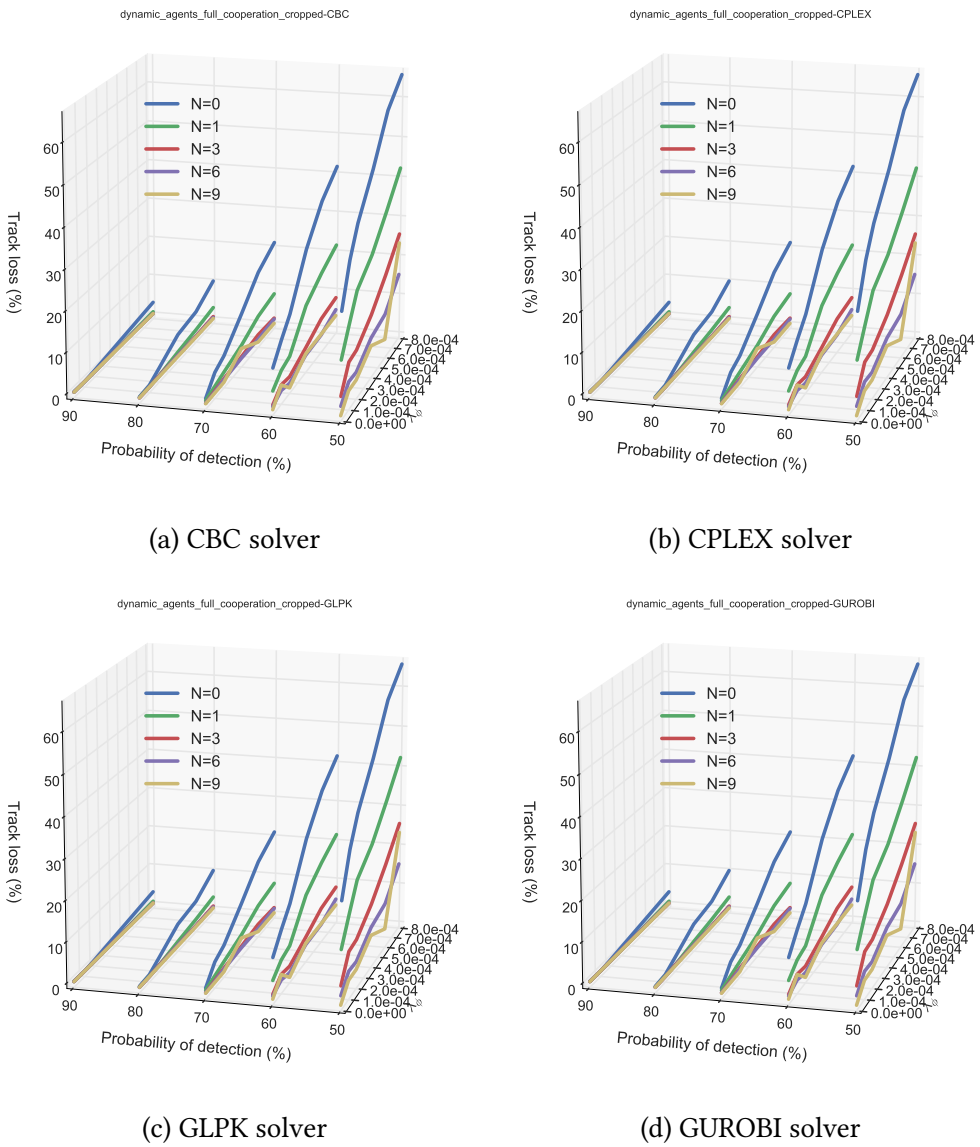
Another improvement that would benefit the run time requirement is to parallelize the algorithm. Since many of the operations in the algorithm are independent, they are excellent candidates for running in parallel. Today CPU's can be delivered with as many as 24 cores per die, and with hyper threading they can process op to 48 threads. With server grade motherboards capable of one to

four CPU's, there is potential to run almost 200 threads if necessary. This would however be a pure brute force approach, which would push the boundary of how many targets the system would be able to track simultaneously, but not the run time as a function of the problem size.

A final but very important piece is also necessary for the system to be complete, track initialization. As with AIS assisted tracking, there are plenty of ways to tackle this challenge. One alternative, and probably the most common for TOMHT is to run a separate system besides the tracking loop to initiate targets. This system can run i.e. recursive RANSAC, PDA-/JPDA-filter, with a selected criterion for birth. The most common is a N out of M threshold, where N and M is about 4 and 6. These separate initialization trackers can run on all measurements, or maybe preferably on the non-active measurements from the primary system.

Appendices

Figures



(a) CBC solver

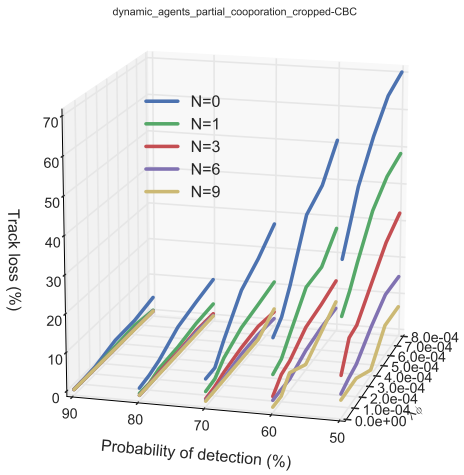
(b) CPLEX solver

(c) GLPK solver

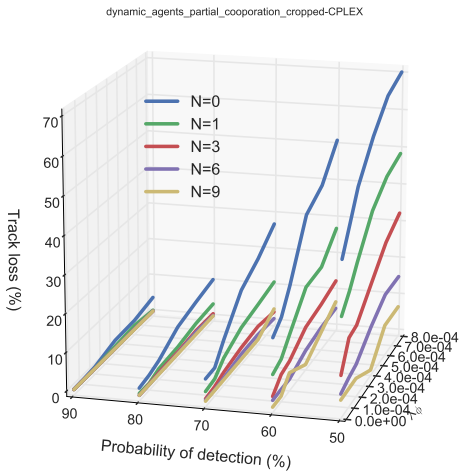
(d) GUROBI solver

Figure 14: Simulation results for all solvers in scenario 1

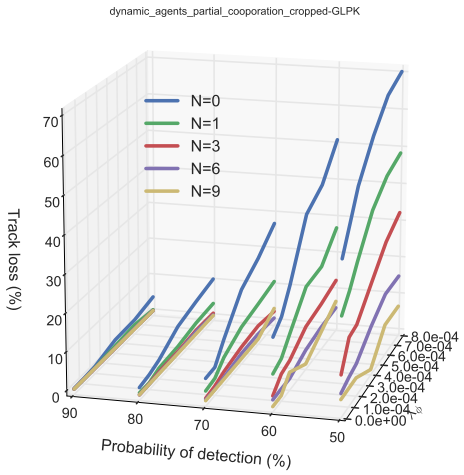
Scenario 2 - Track performance



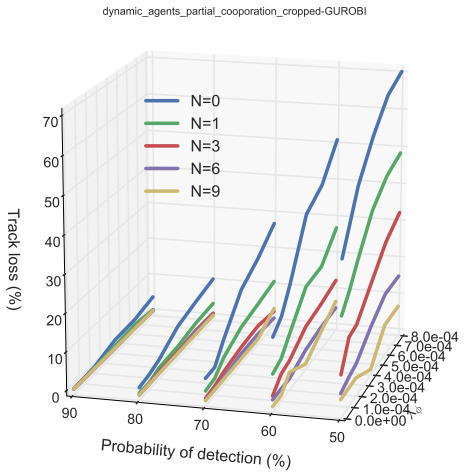
(a) CBC solver



(b) CPLEX solver



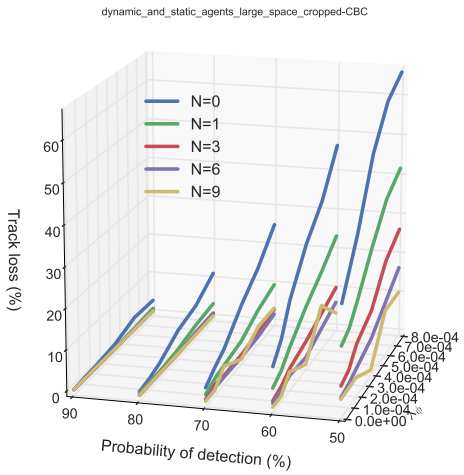
(c) GLPK solver



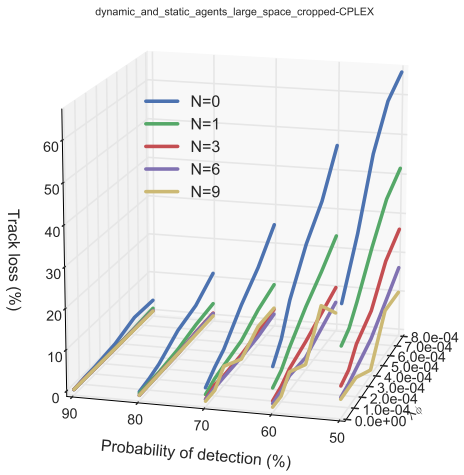
(d) GUROBI solver

Figure 15: Simulation results for all solvers in scenario 2

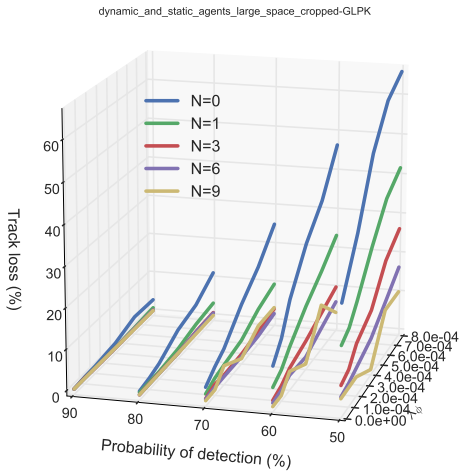
Scenario3 - Track performance



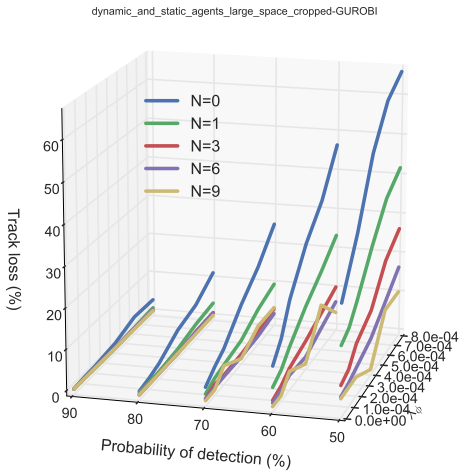
(a) CBC solver



(b) CPLEX solver



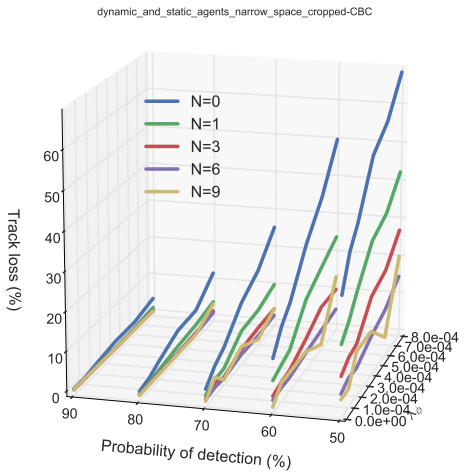
(c) GLPK solver



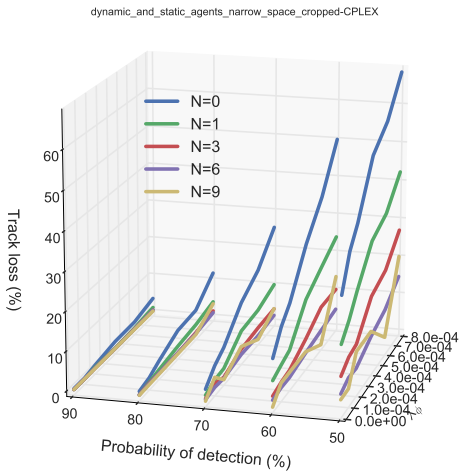
(d) GUROBI solver

Figure 16: Simulation results for all solvers in scenario 3

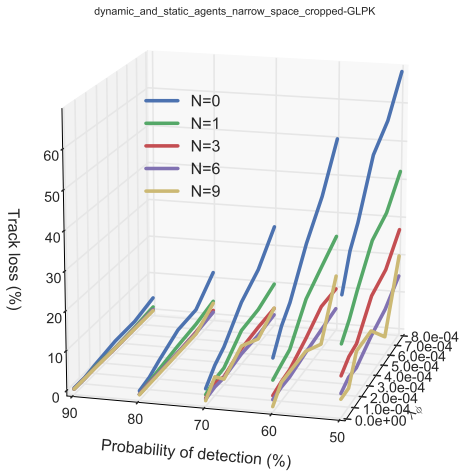
Scenario 4 - Track performance



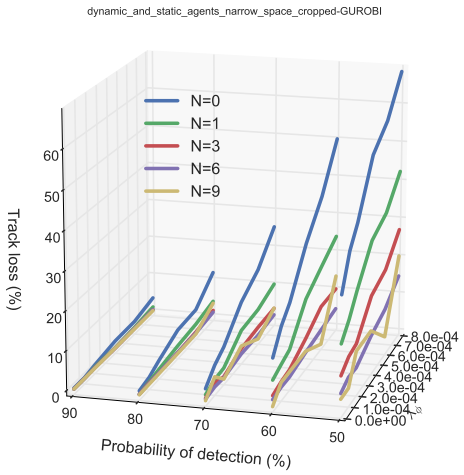
(a) CBC solver



(b) CPLEX solver



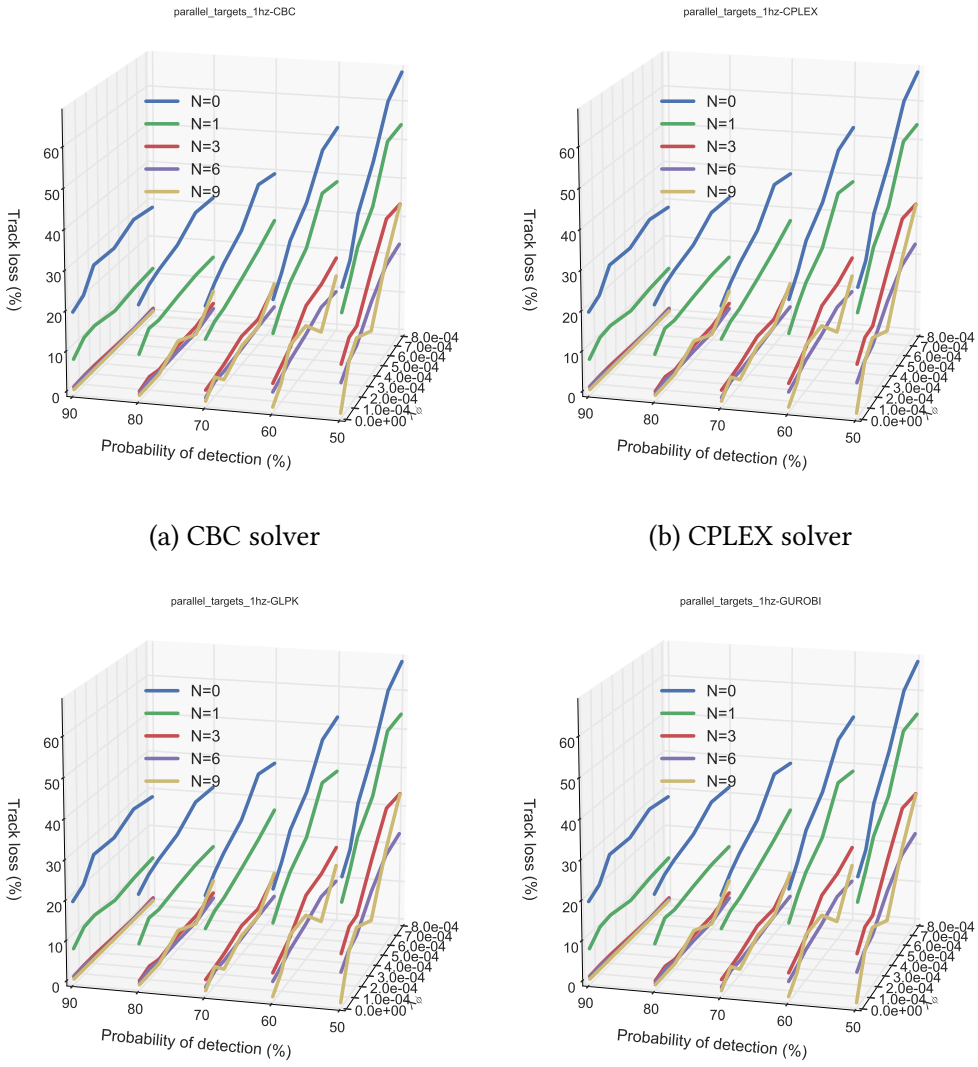
(c) GLPK solver



(d) GUROBI solver

Figure 17: Simulation results for all solvers in scenario 4

Scenario 5 - Track performance



(a) CBC solver

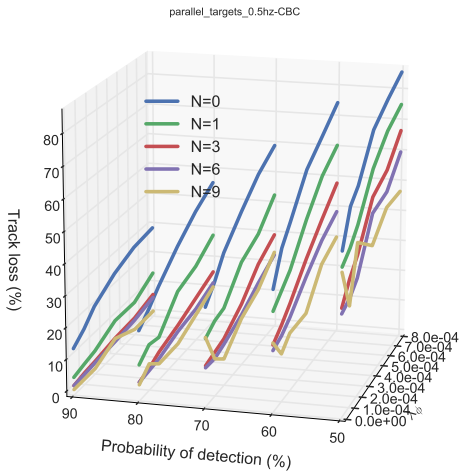
(b) CPLEX solver

(c) GLPK solver

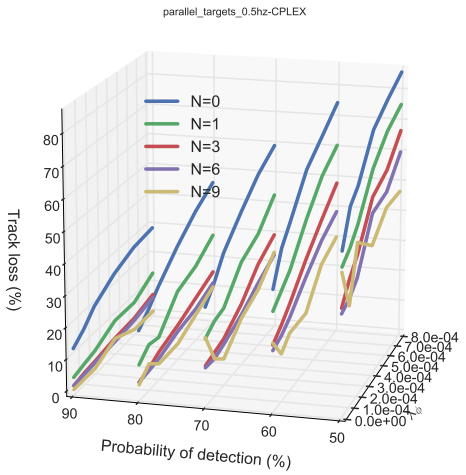
(d) GUROBI solver

Figure 18: Simulation results for all solvers in scenario 5

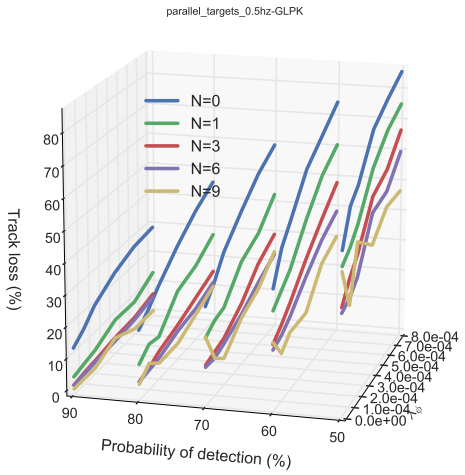
Scenario 6 - Track performance



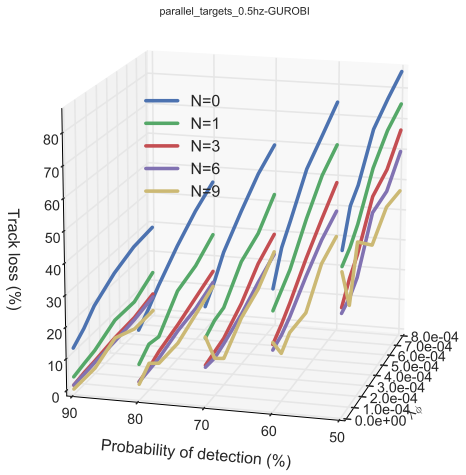
(a) CBC solver



(b) CPLEX solver



(c) GLPK solver



(d) GUROBI solver

Figure 19: Simulation results for all solvers in scenario 6

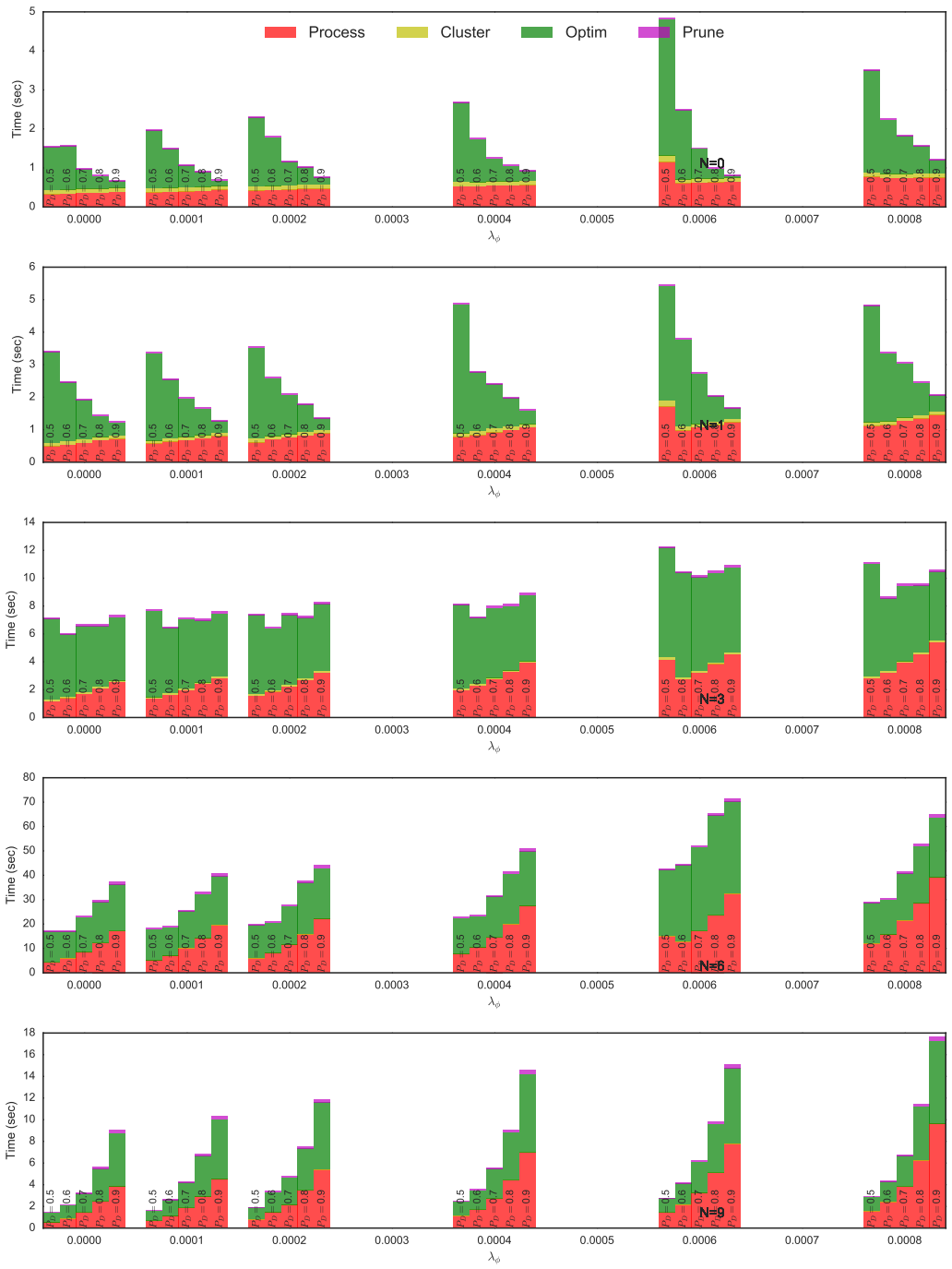


Figure 20: Scenario 1 - Run time log, CBC solver

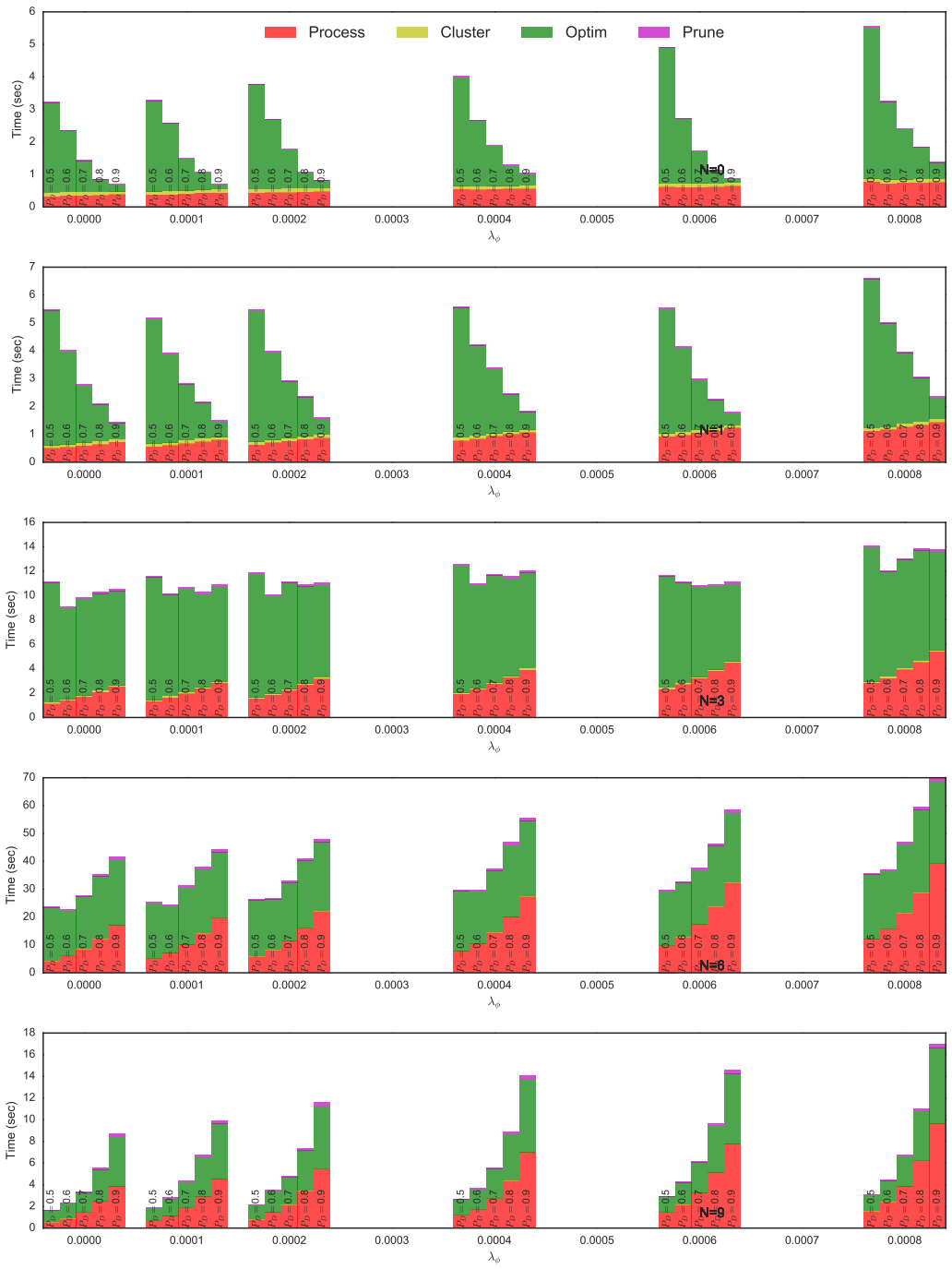


Figure 21: Scenario 1 - Run time log, CPLEX solver

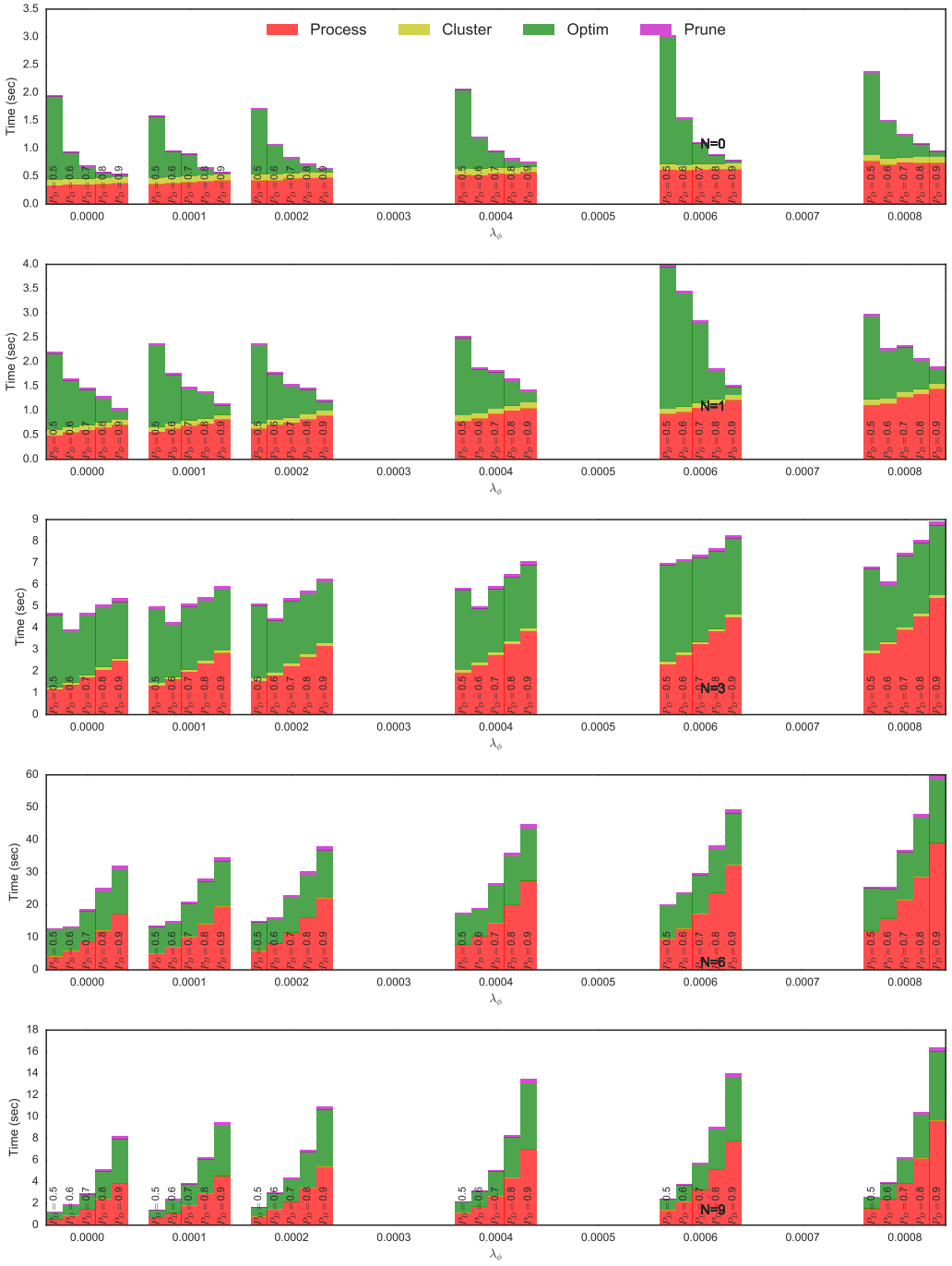


Figure 22: Scenario 1 - Run time log, GLPK solver

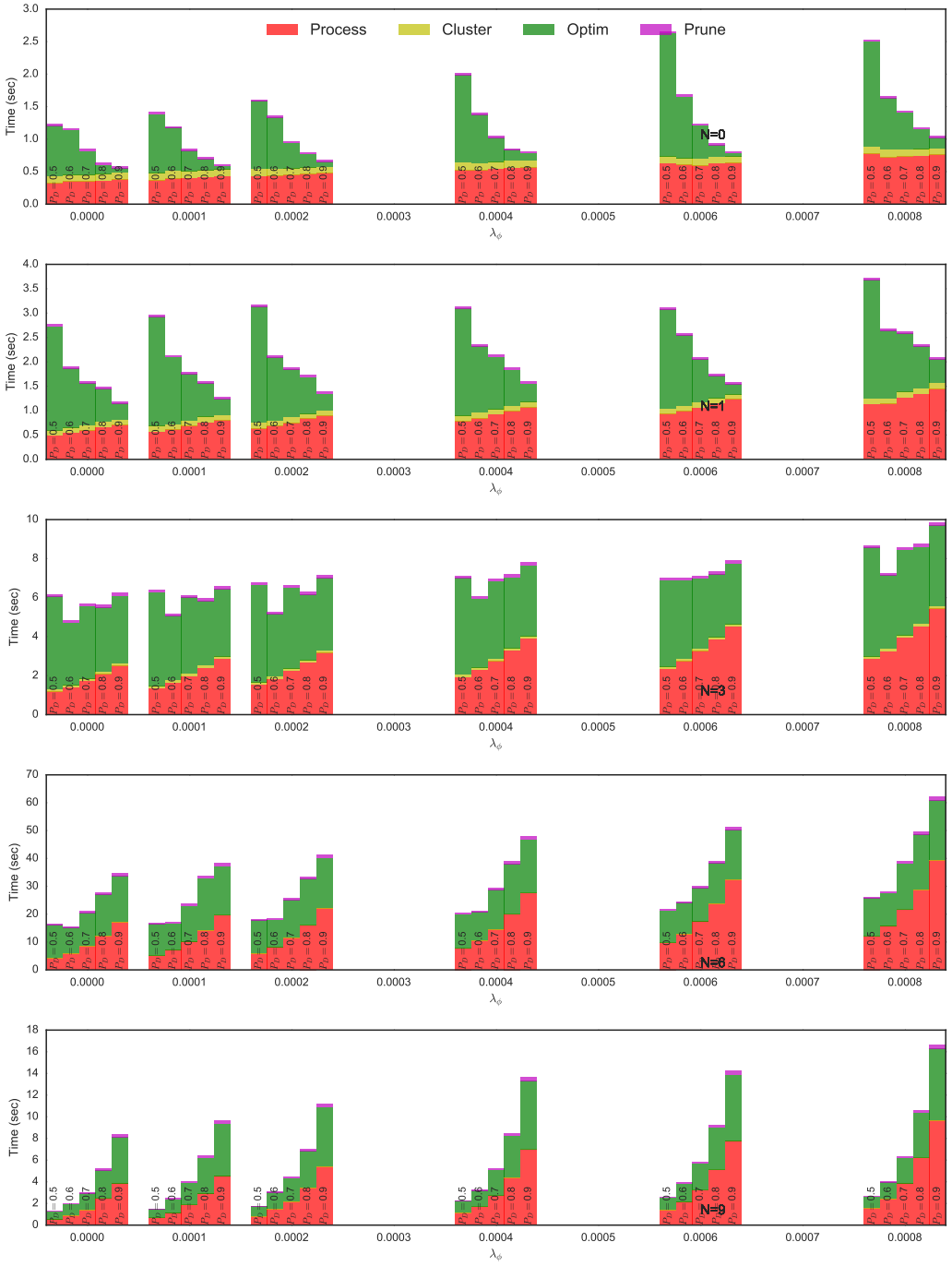


Figure 23: Scenario 1 - Run time log, GUROBI solver

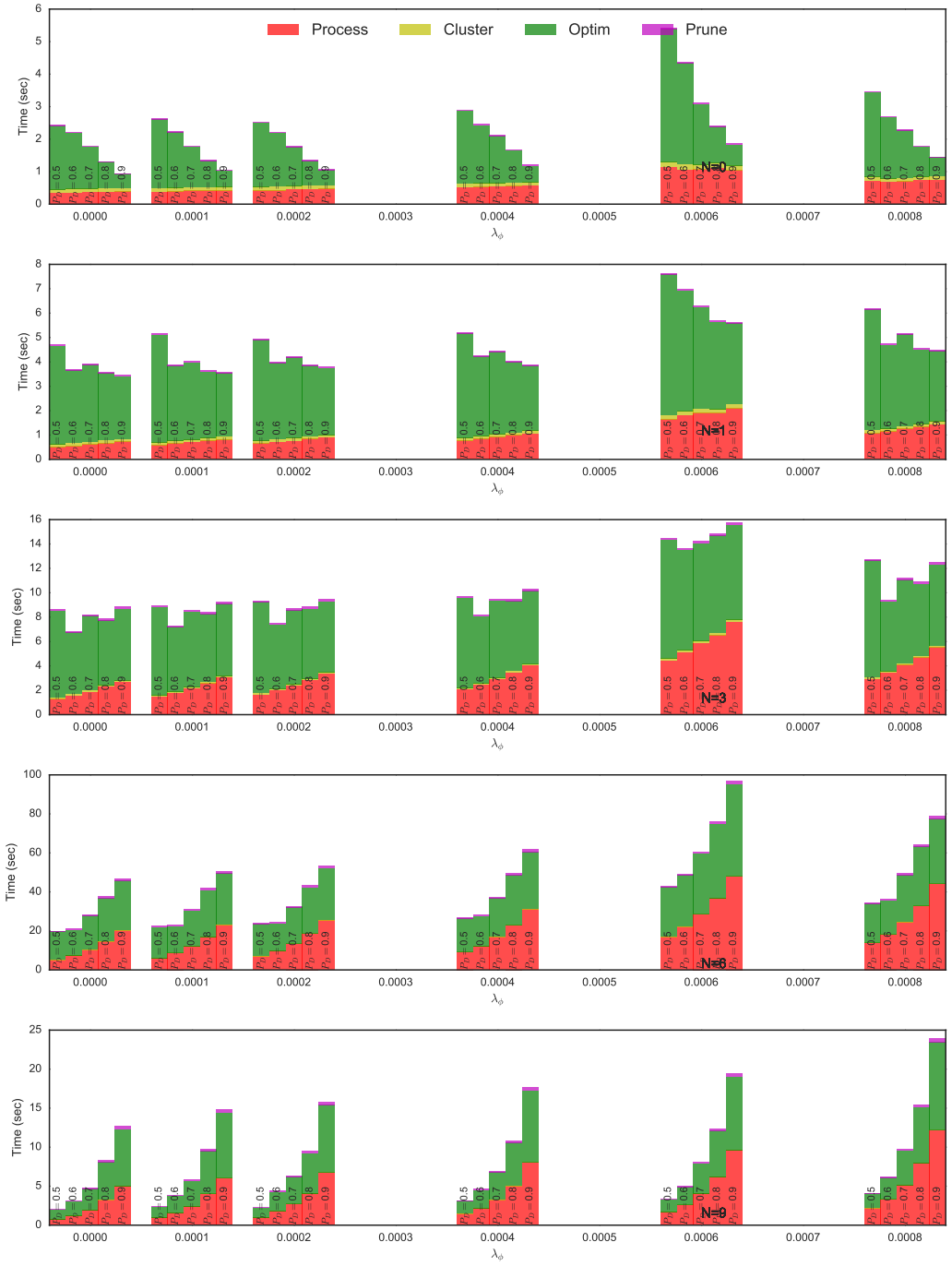


Figure 24: Scenario 2 - Run time log, CBC solver

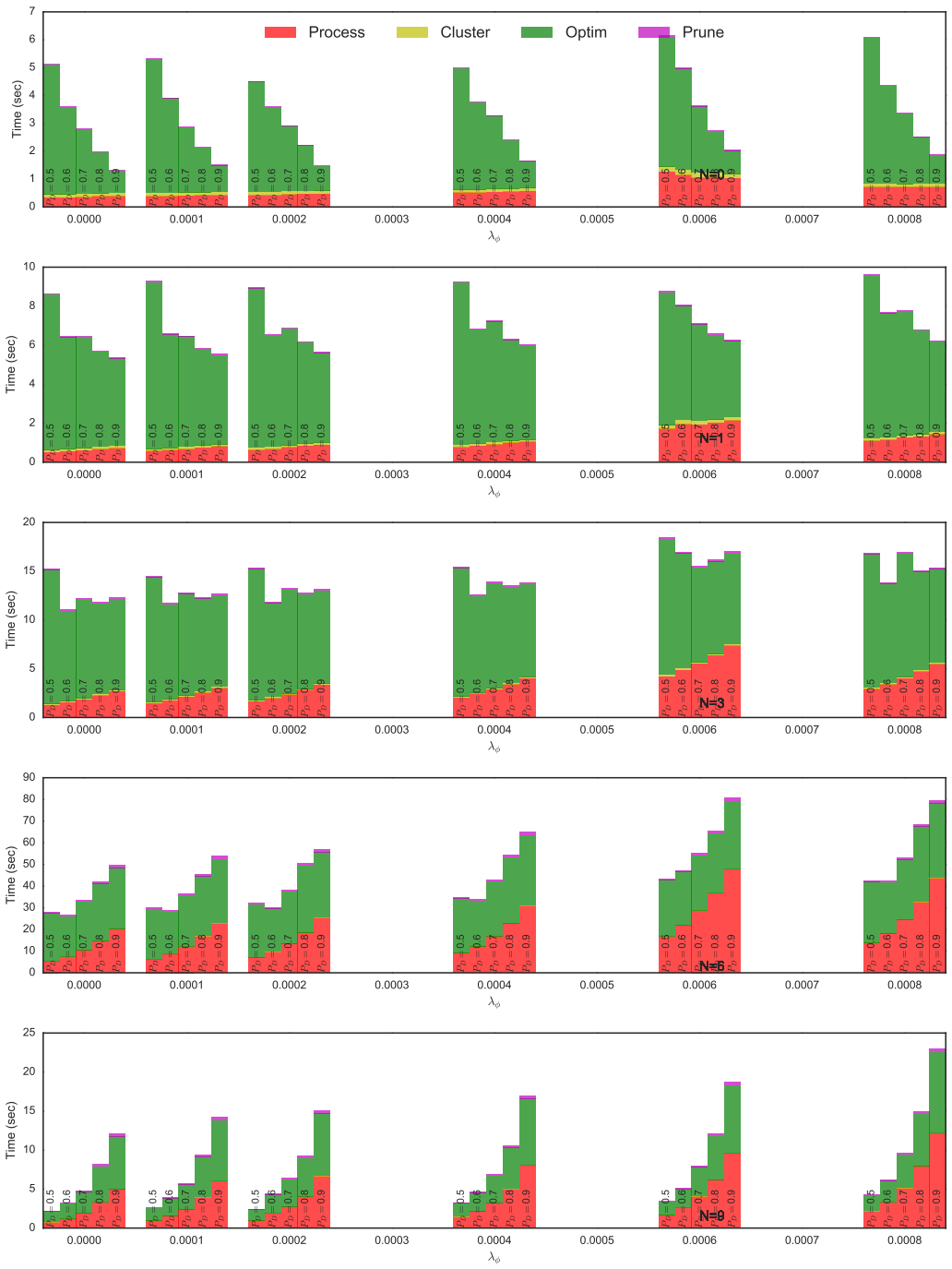


Figure 25: Scenario 2 - Run time log, CPLEX solver

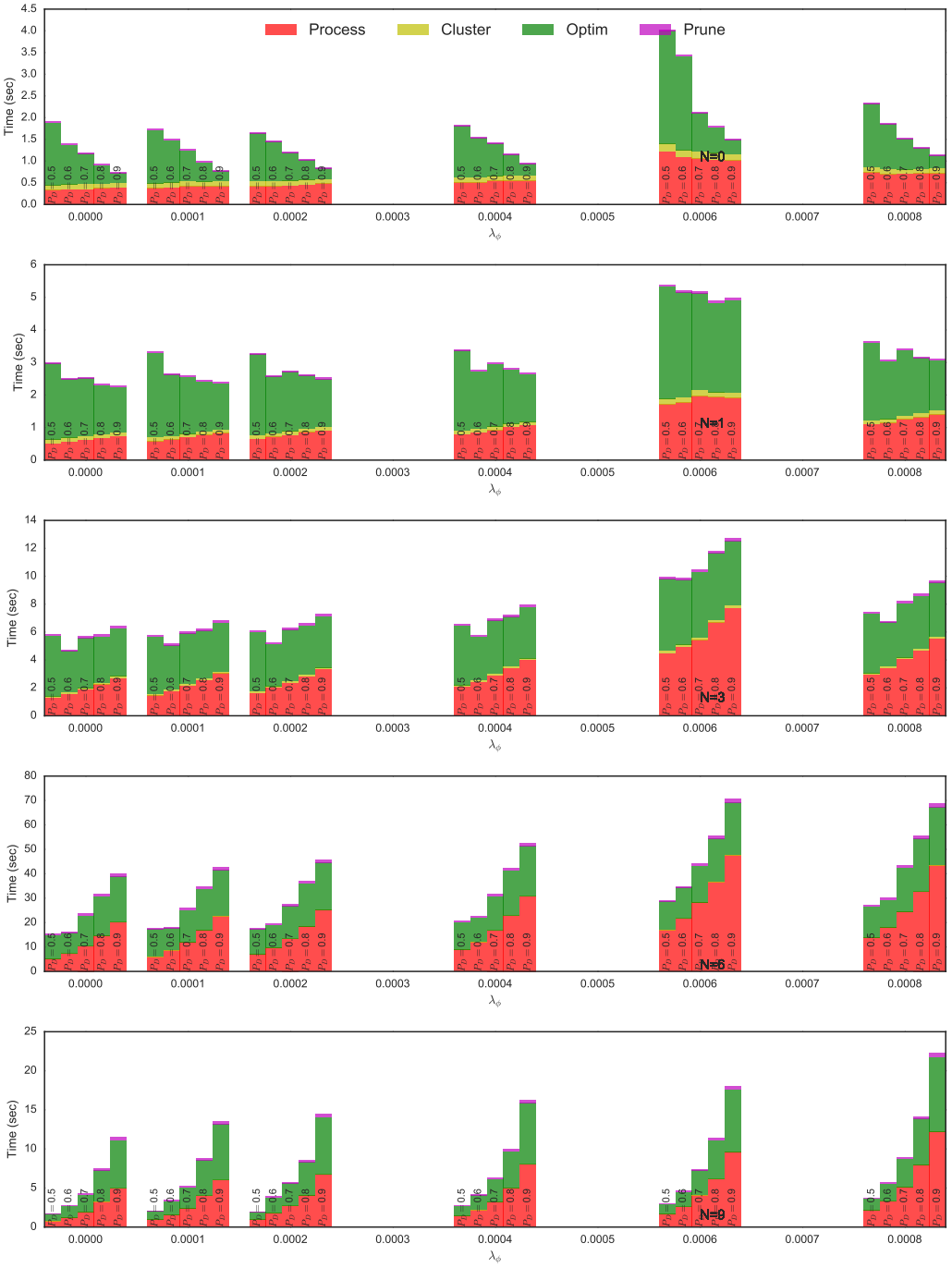


Figure 26: Scenario 2 - Run time log, GLPK solver

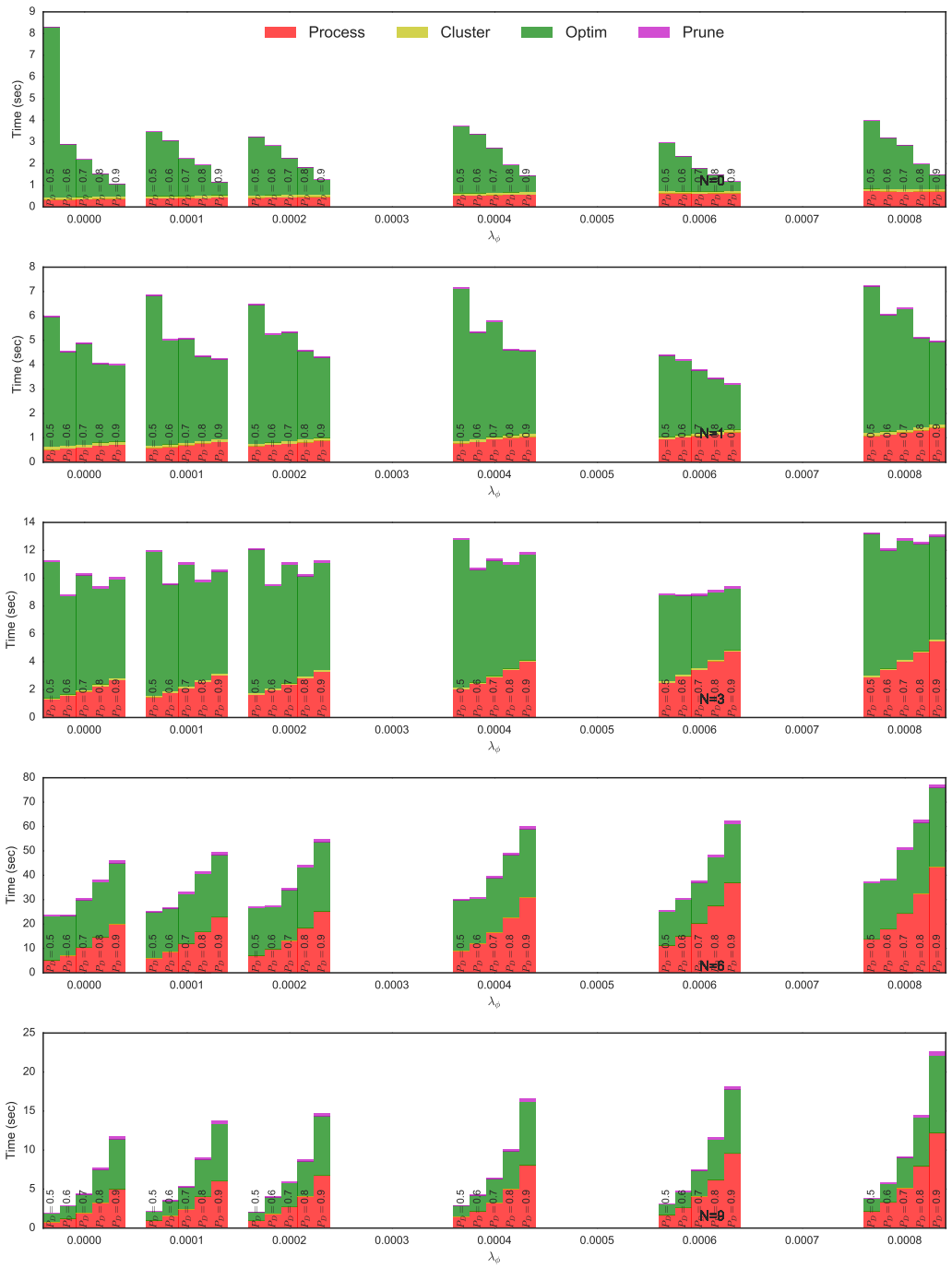


Figure 27: Scenario 2 - Run time log, GUROBI solver

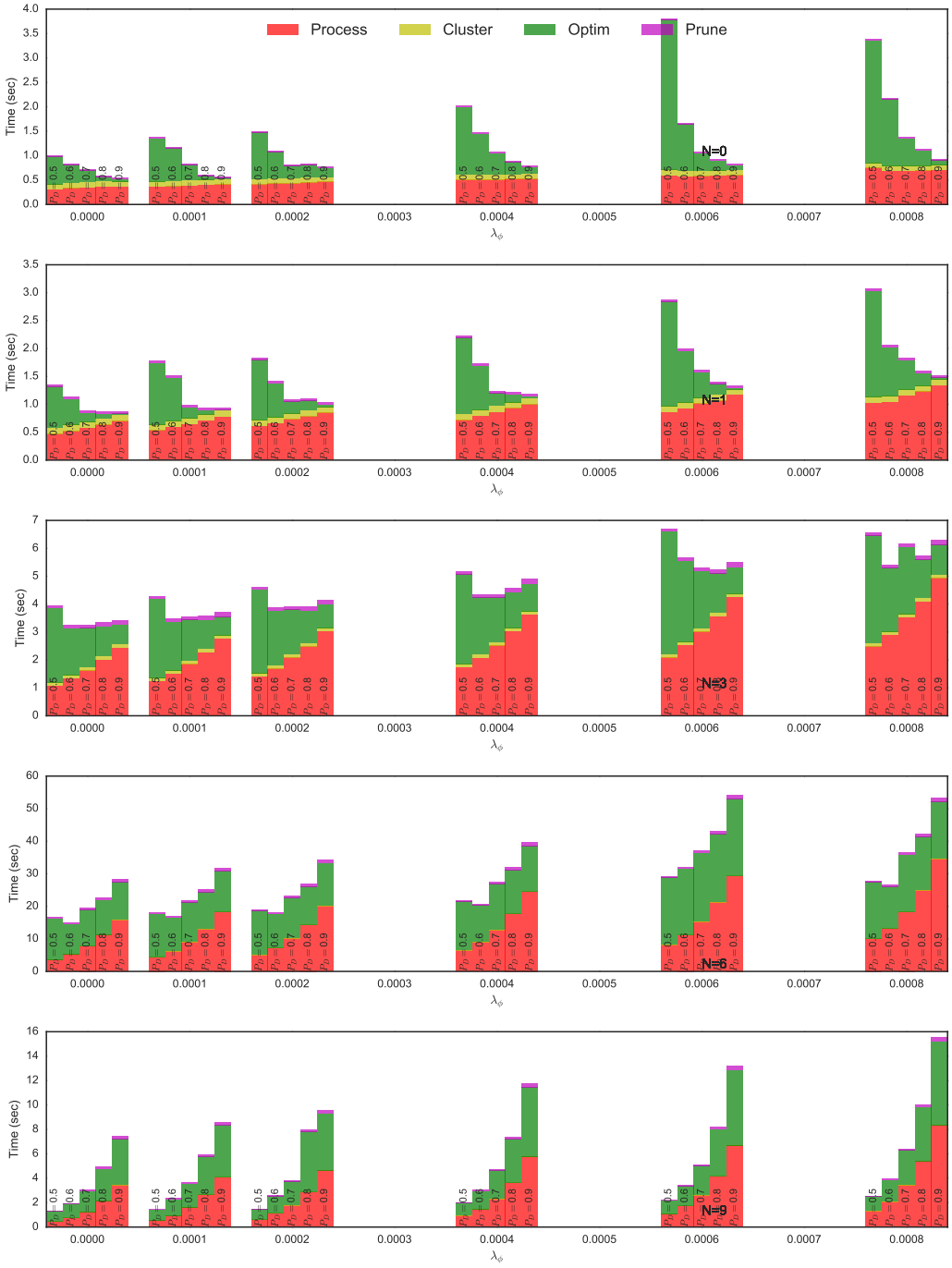


Figure 28: Scenario3 - Run time log, CBC solver

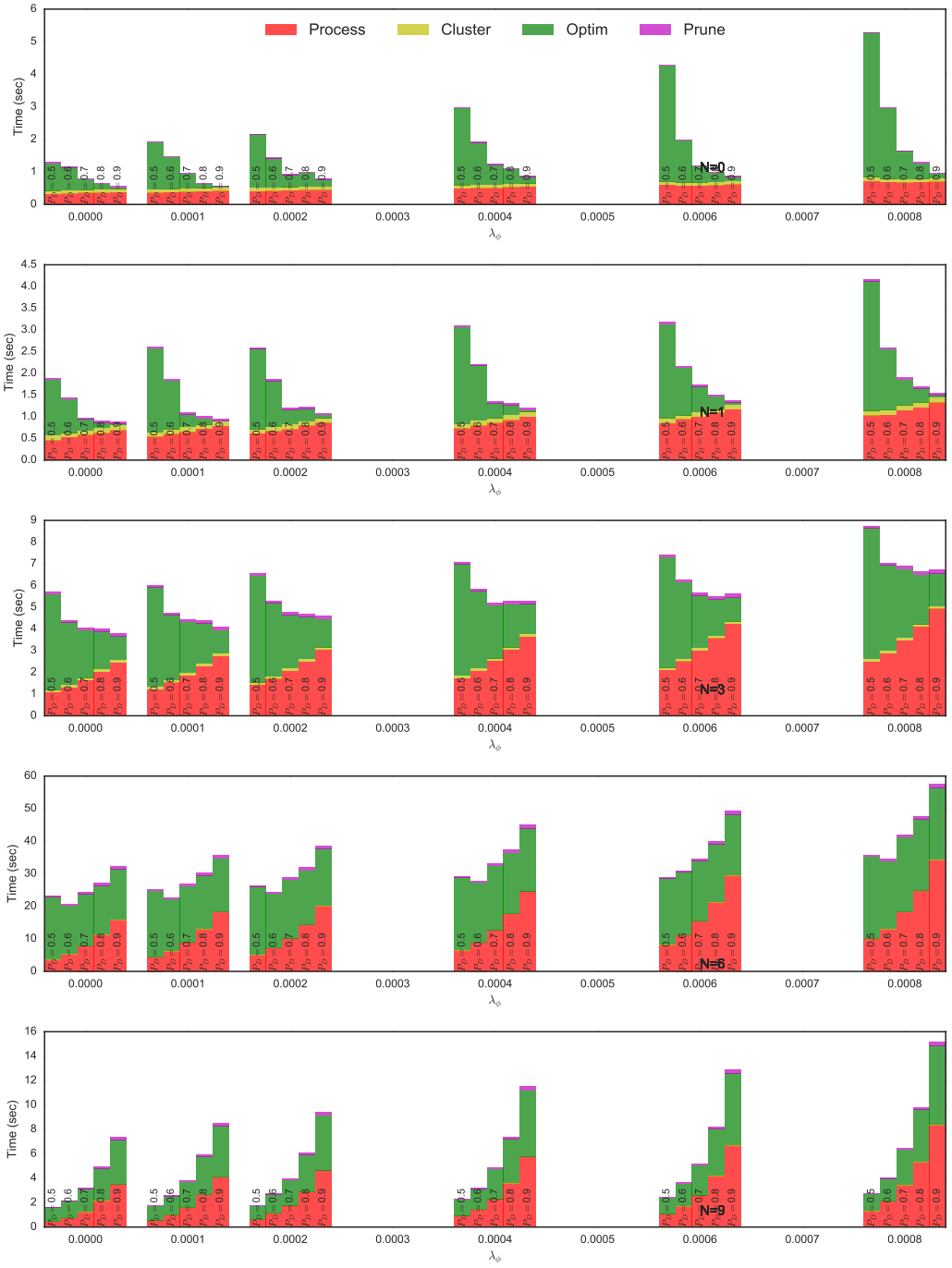


Figure 29: Scenario3 - Run time log, CPLEX solver

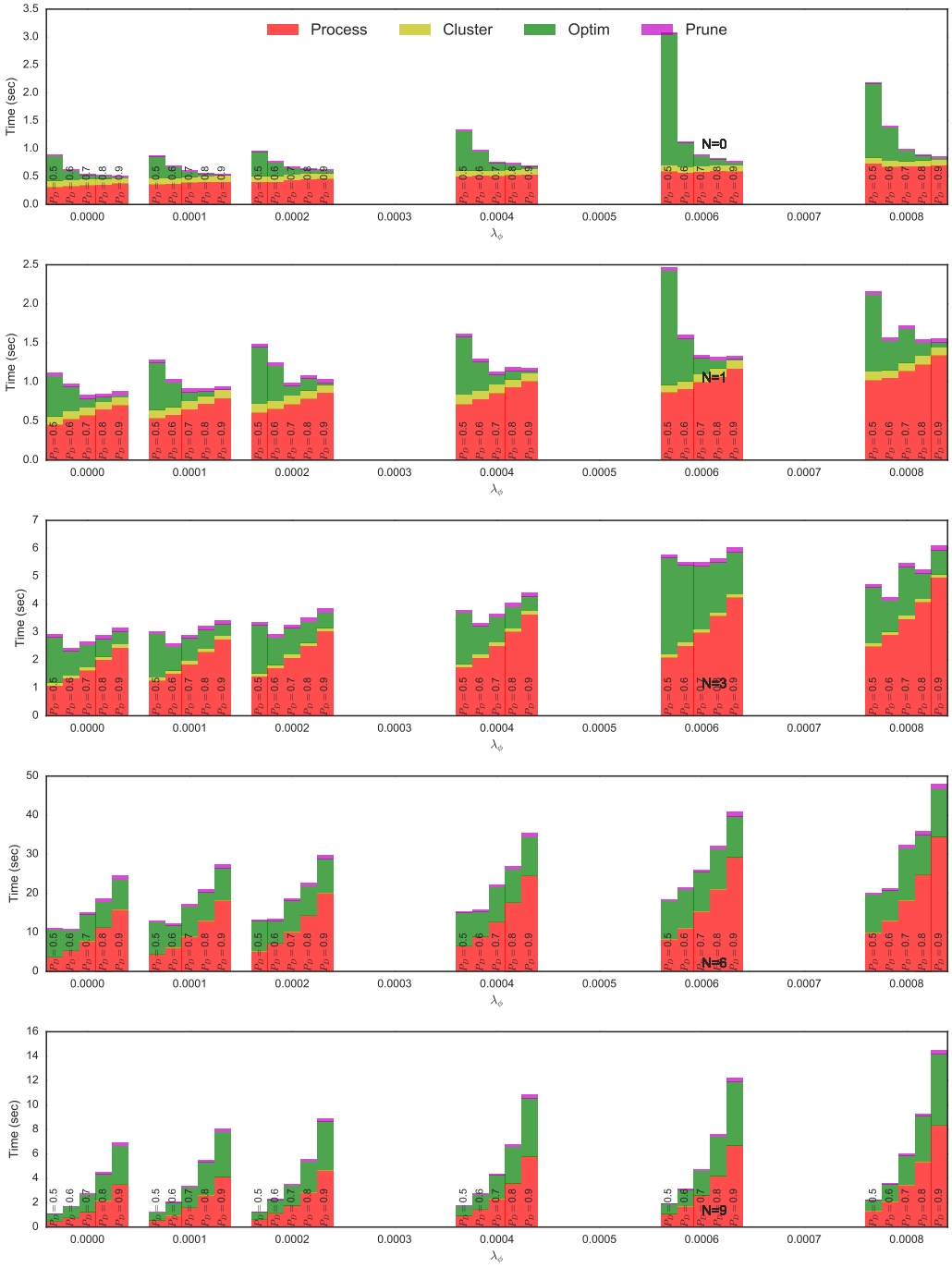


Figure 30: Scenario3 - Run time log, GLPK solver

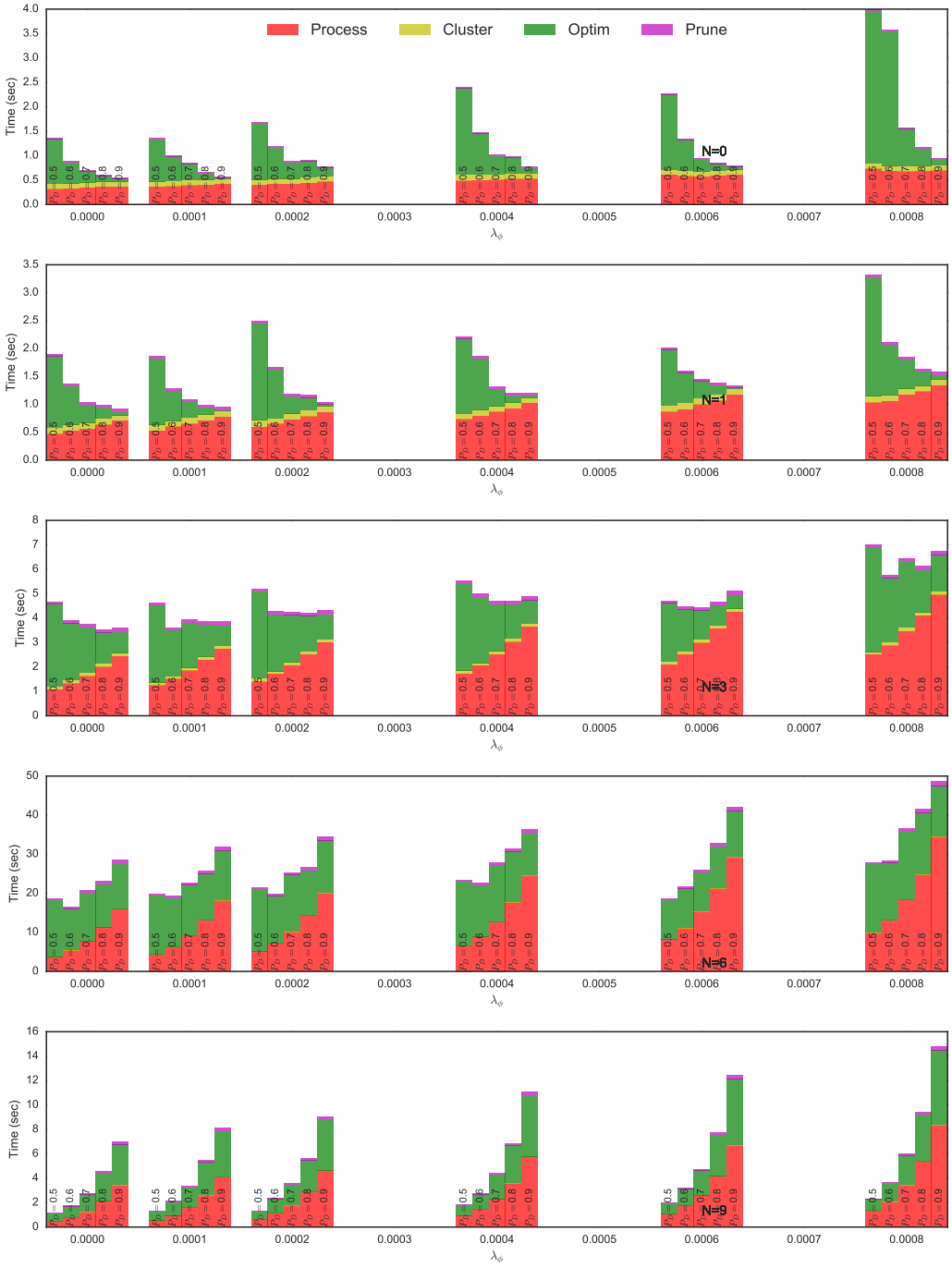


Figure 31: Scenario3 - Run time log, GUROBI solver

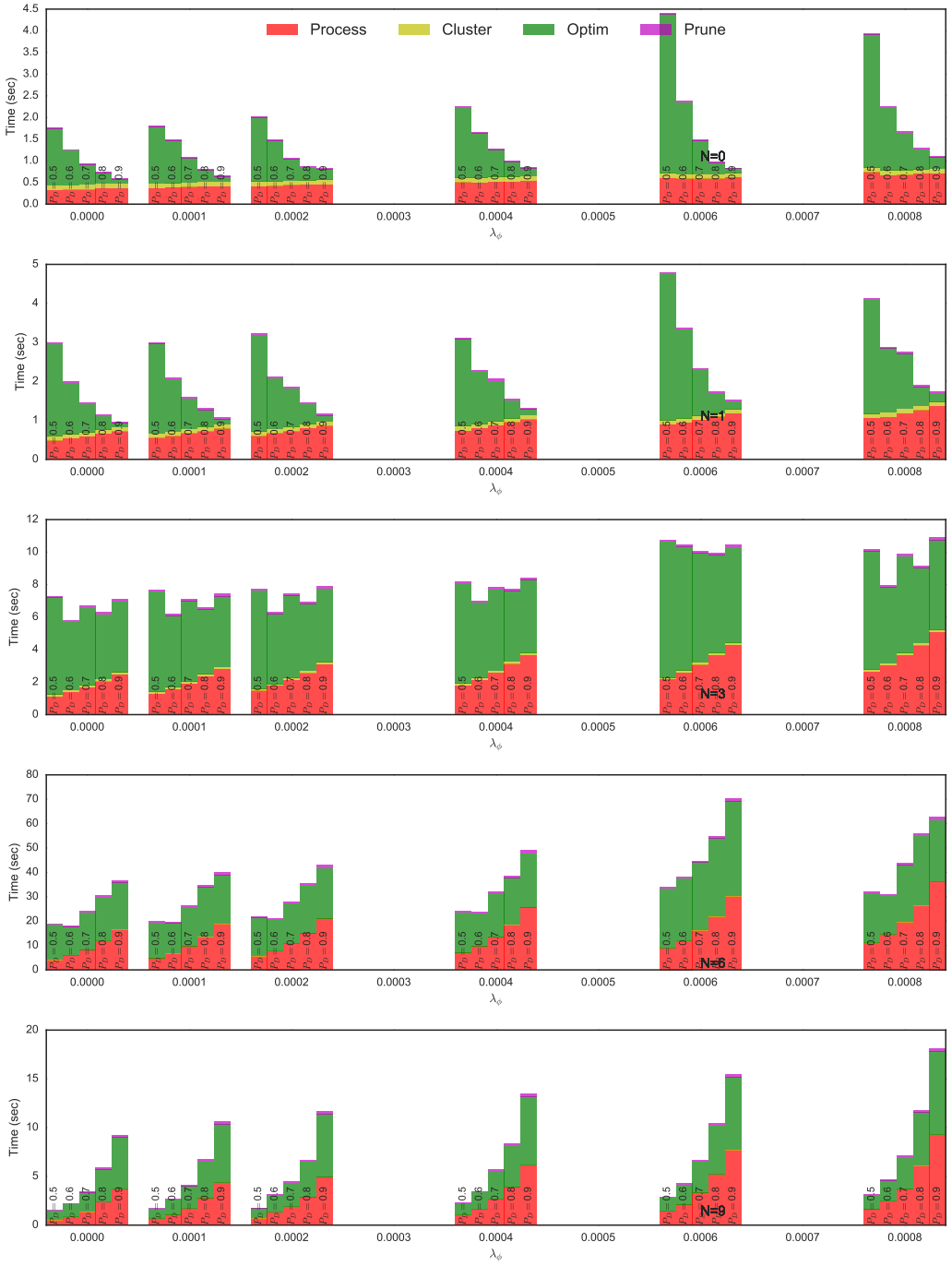


Figure 32: Scenario 4 - Run time log, CBC solver

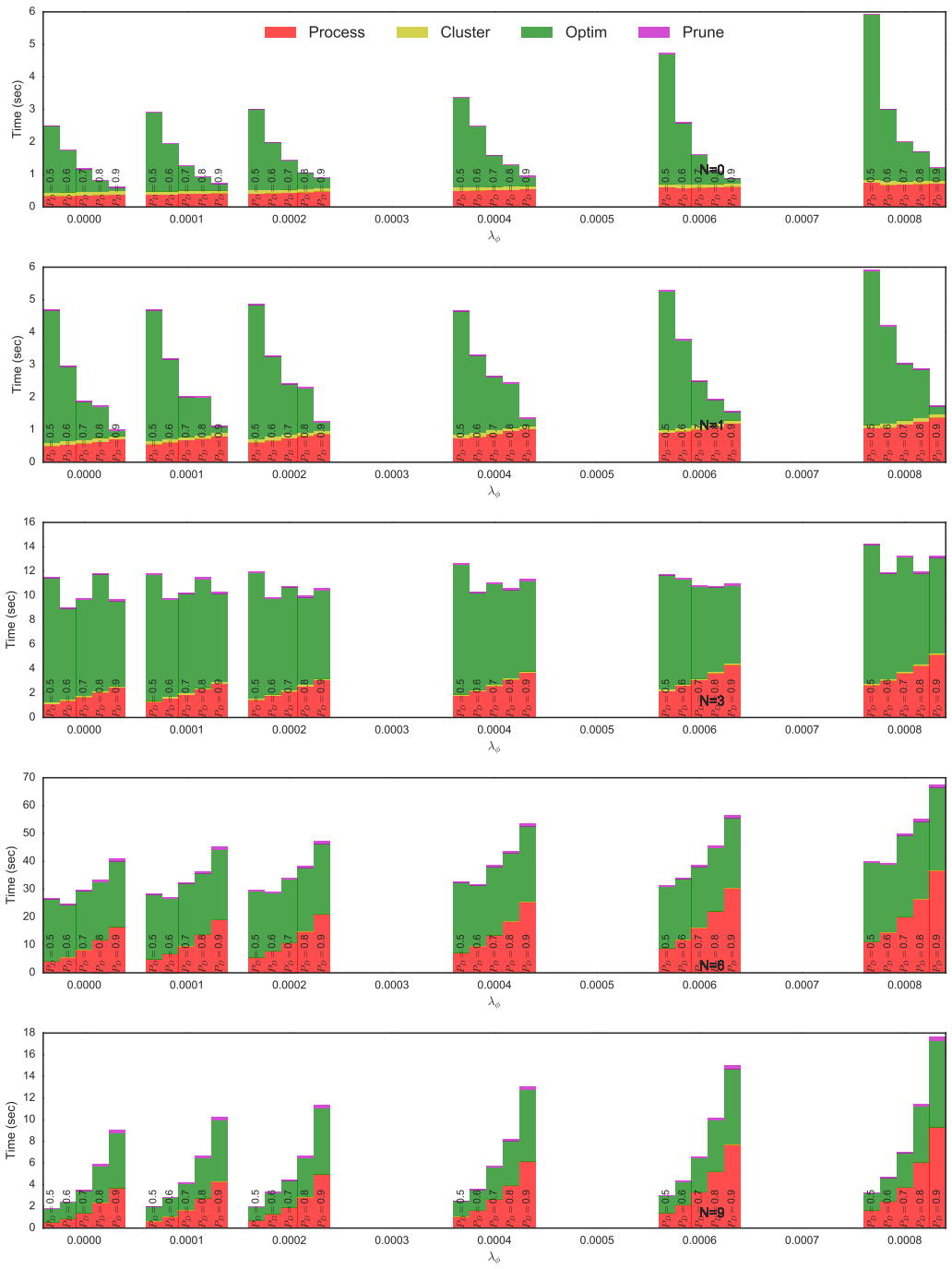


Figure 33: Scenario 4 - Run time log, CPLEX solver

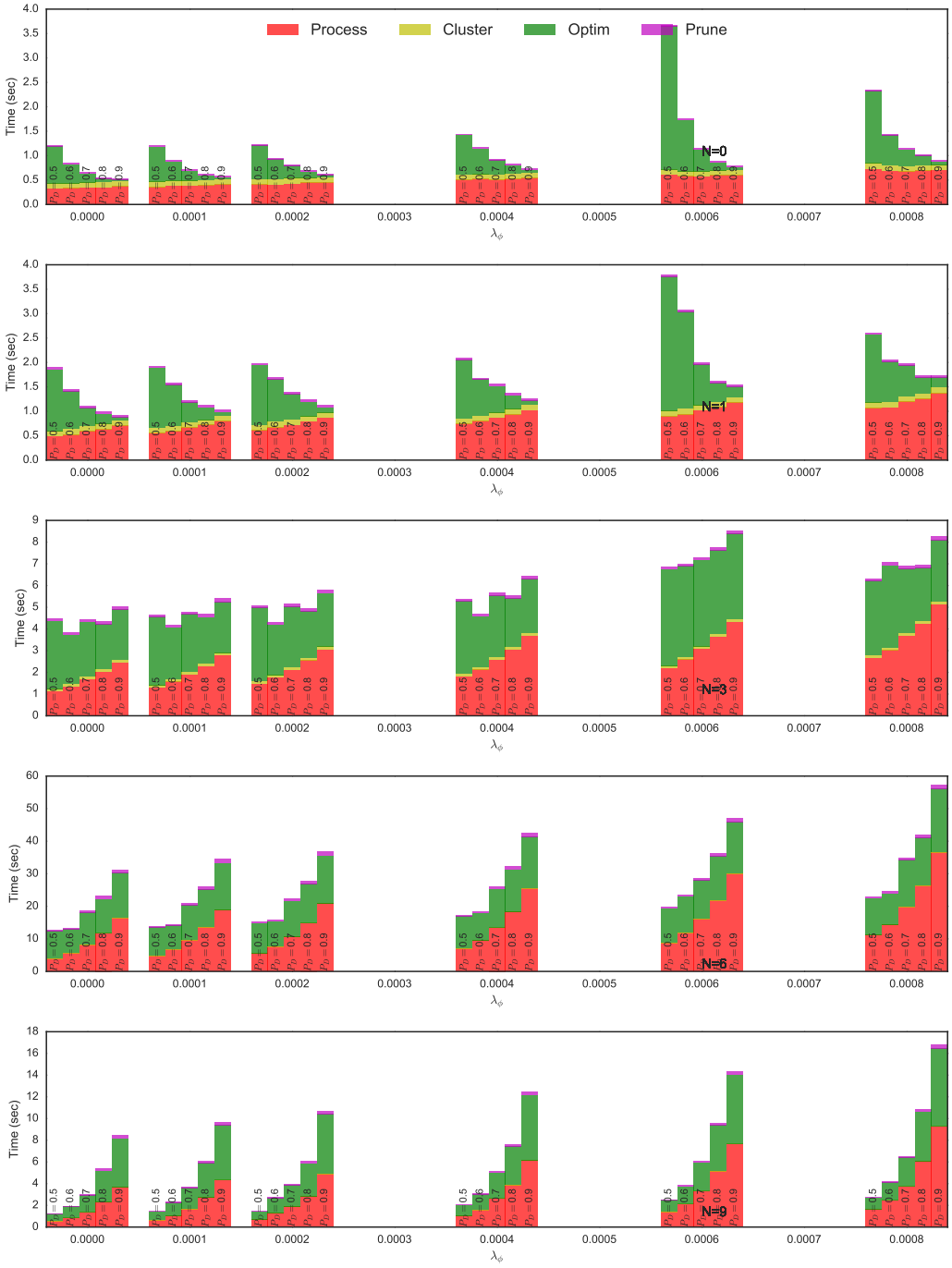


Figure 34: Scenario 4 - Run time log, GLPK solver

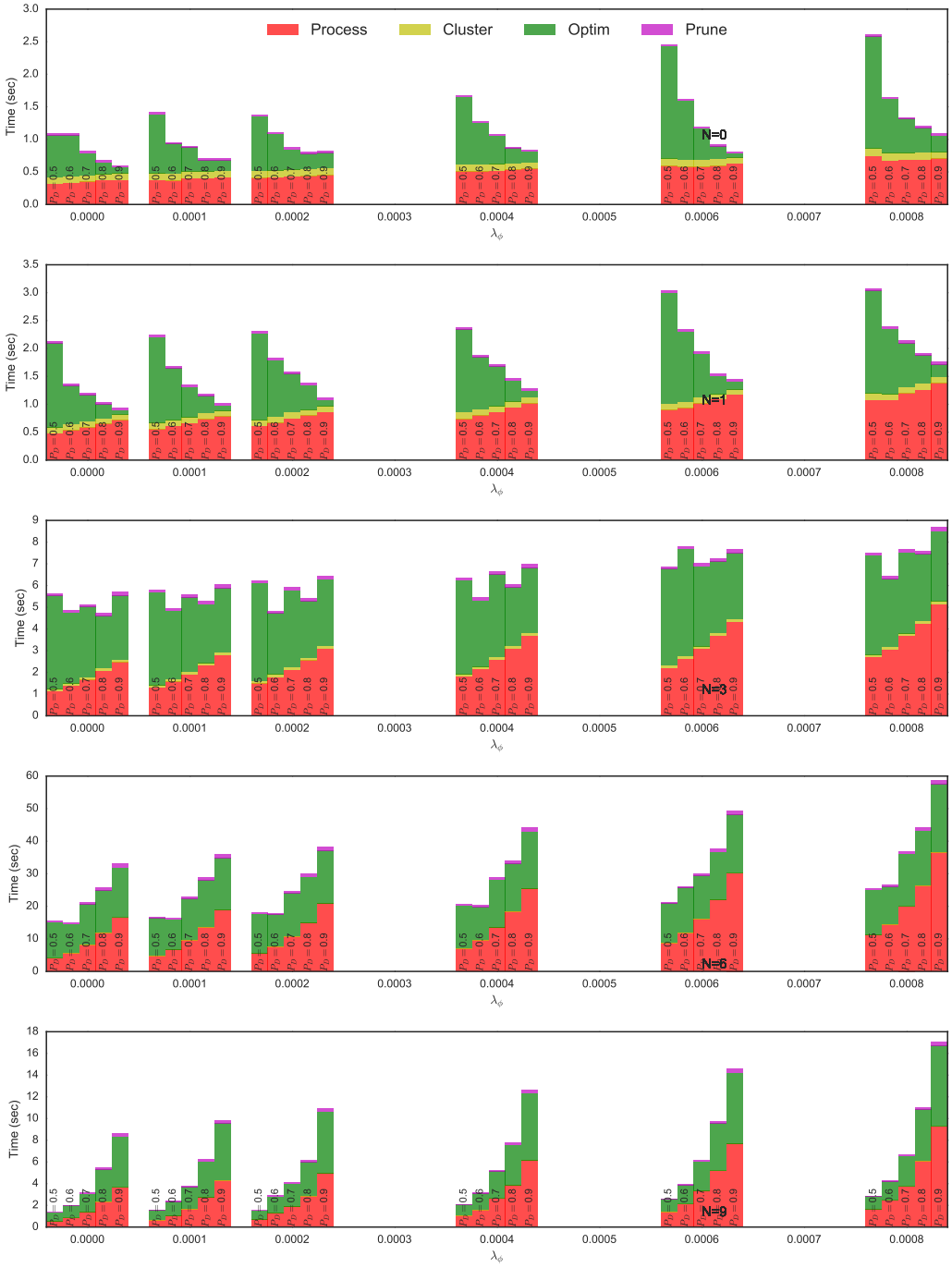


Figure 35: Scenario 4 - Run time log, GUROBI solver

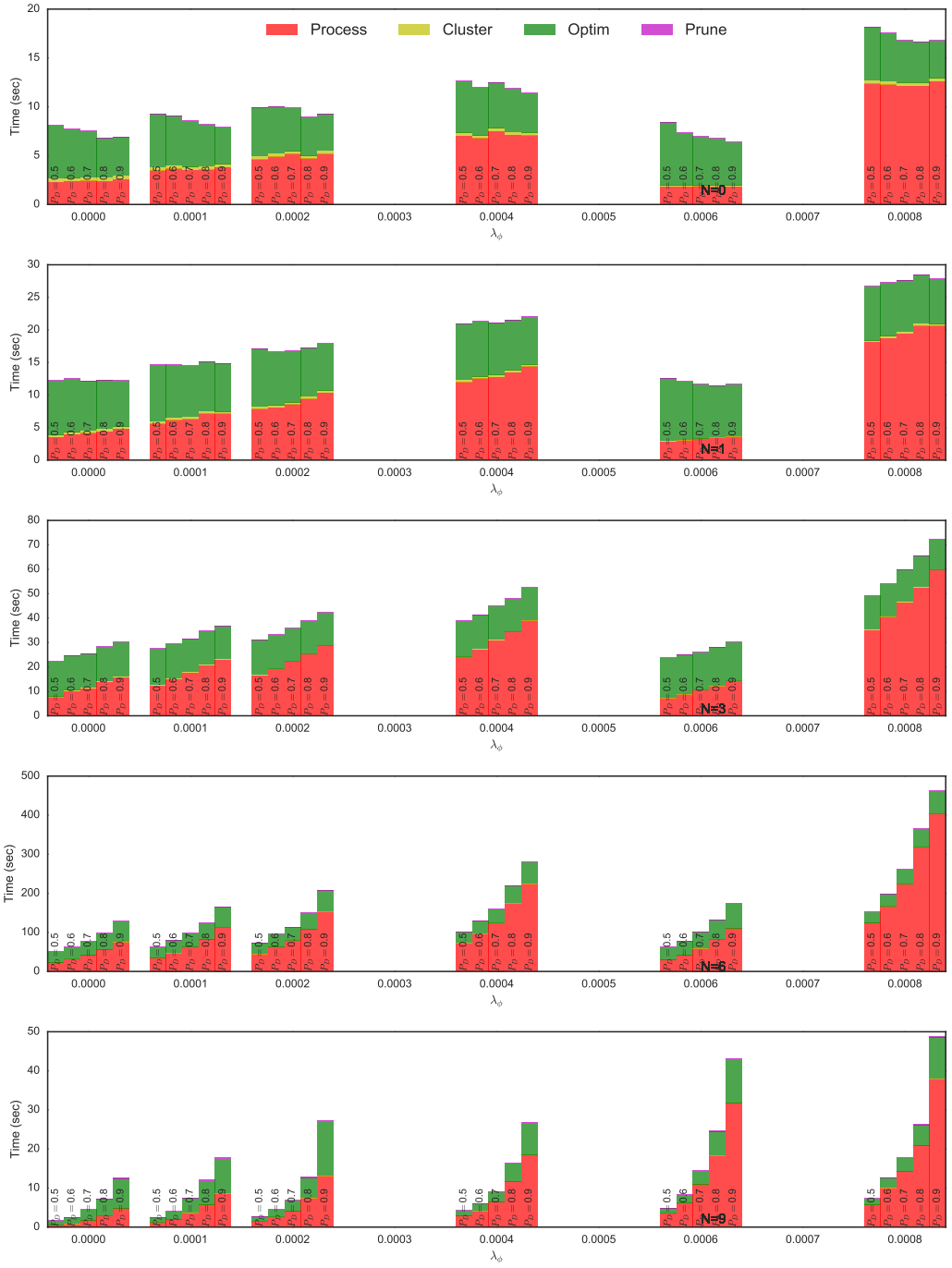


Figure 36: Scenario 5 - Run time log, CBC solver

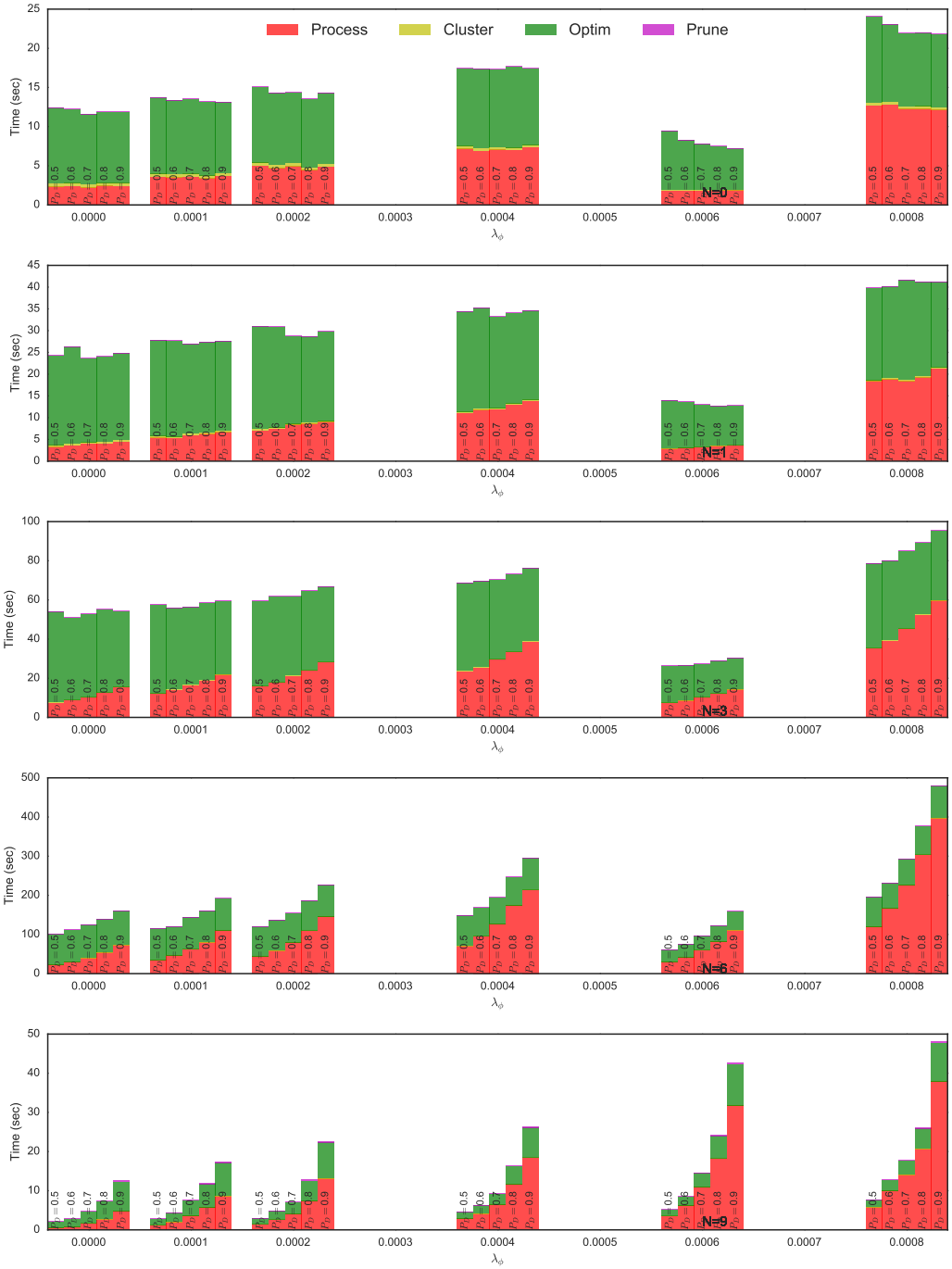


Figure 37: Scenario 5 - Run time log, CPLEX solver

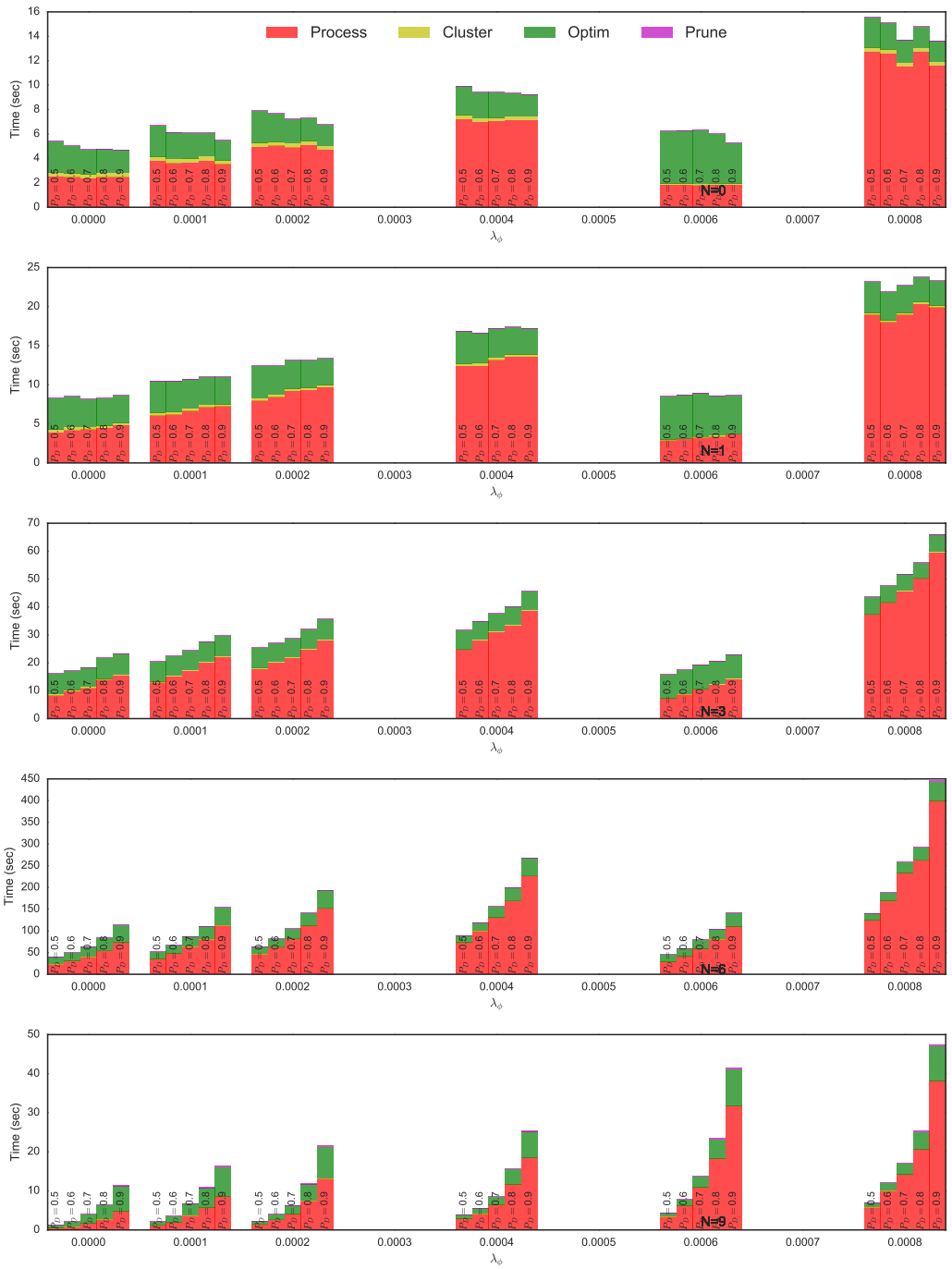


Figure 38: Scenario 5 - Run time log, GLPK solver

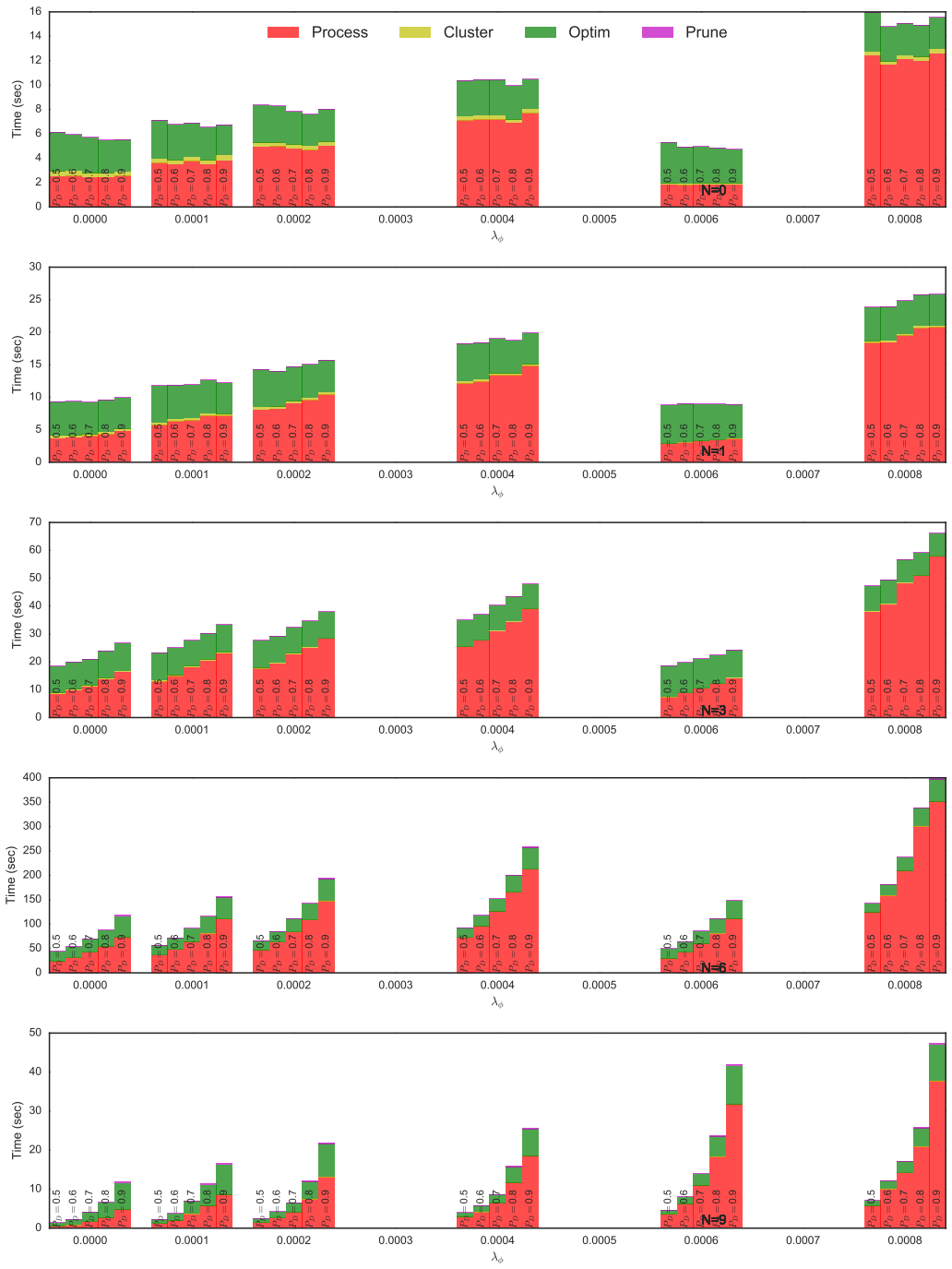


Figure 39: Scenario 5 - Run time log, GUROBI solver

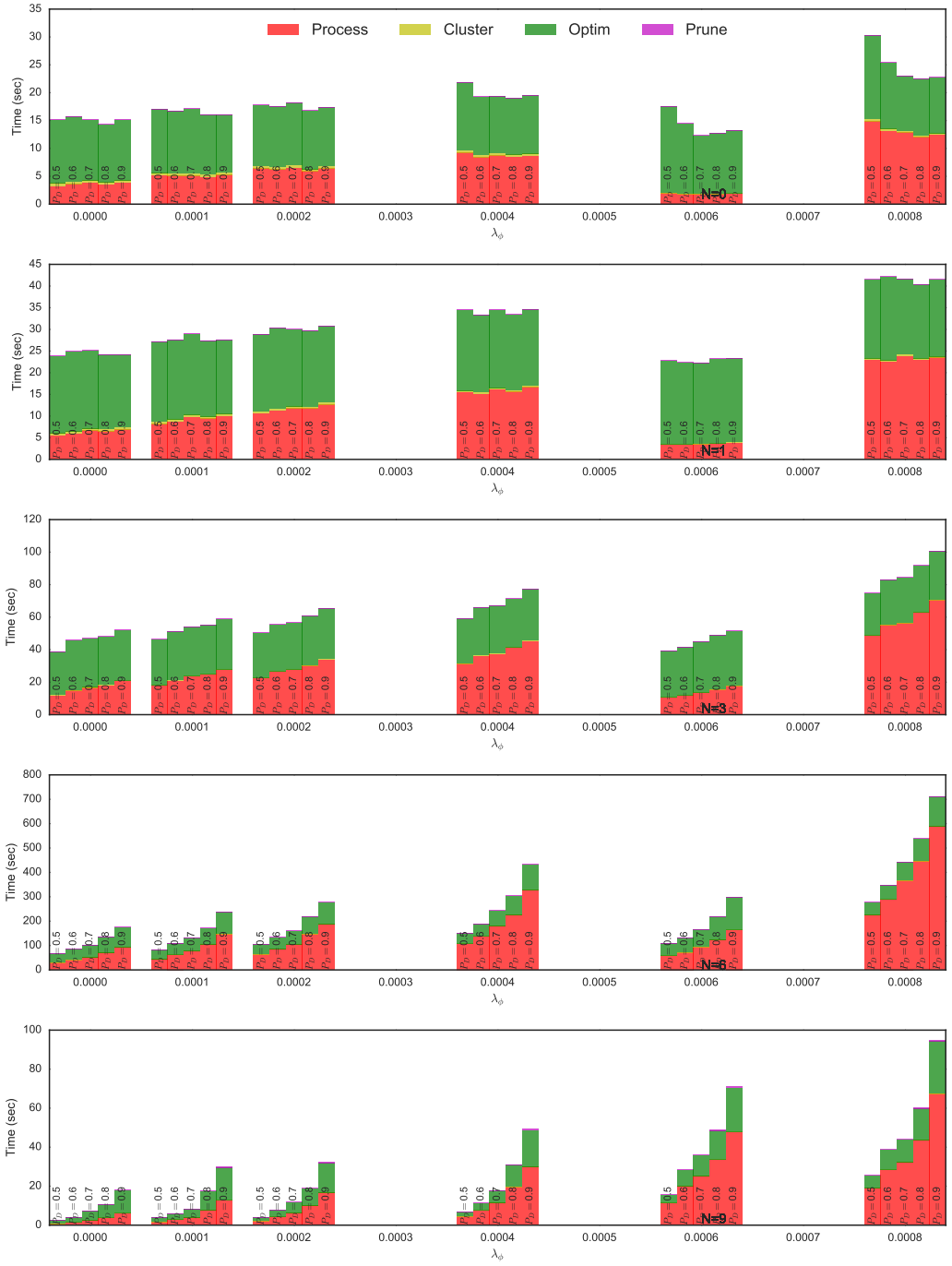


Figure 40: Scenario 6 - Run time log, CBC solver

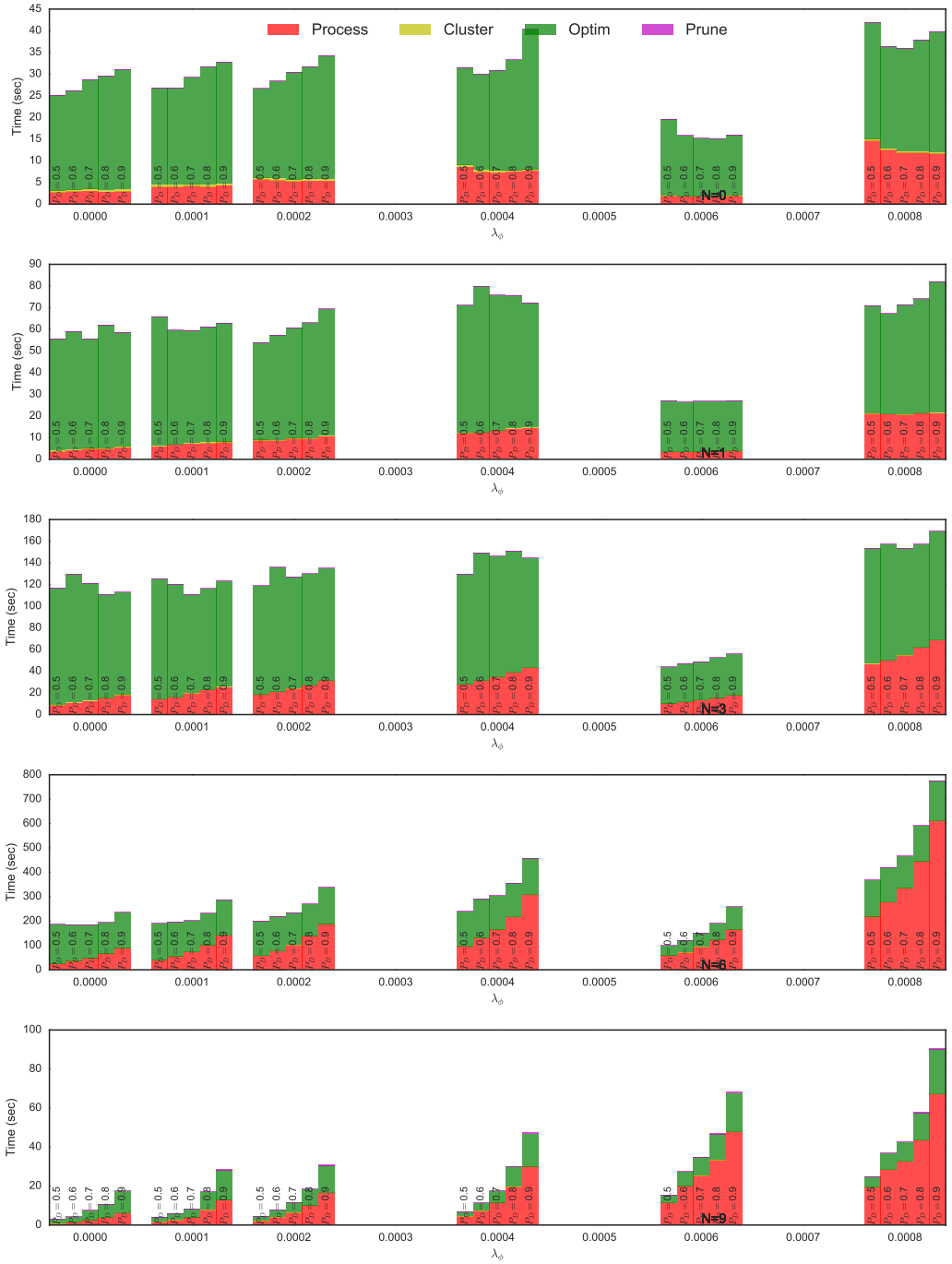


Figure 41: Scenario 6 - Run time log, CPLEX solver

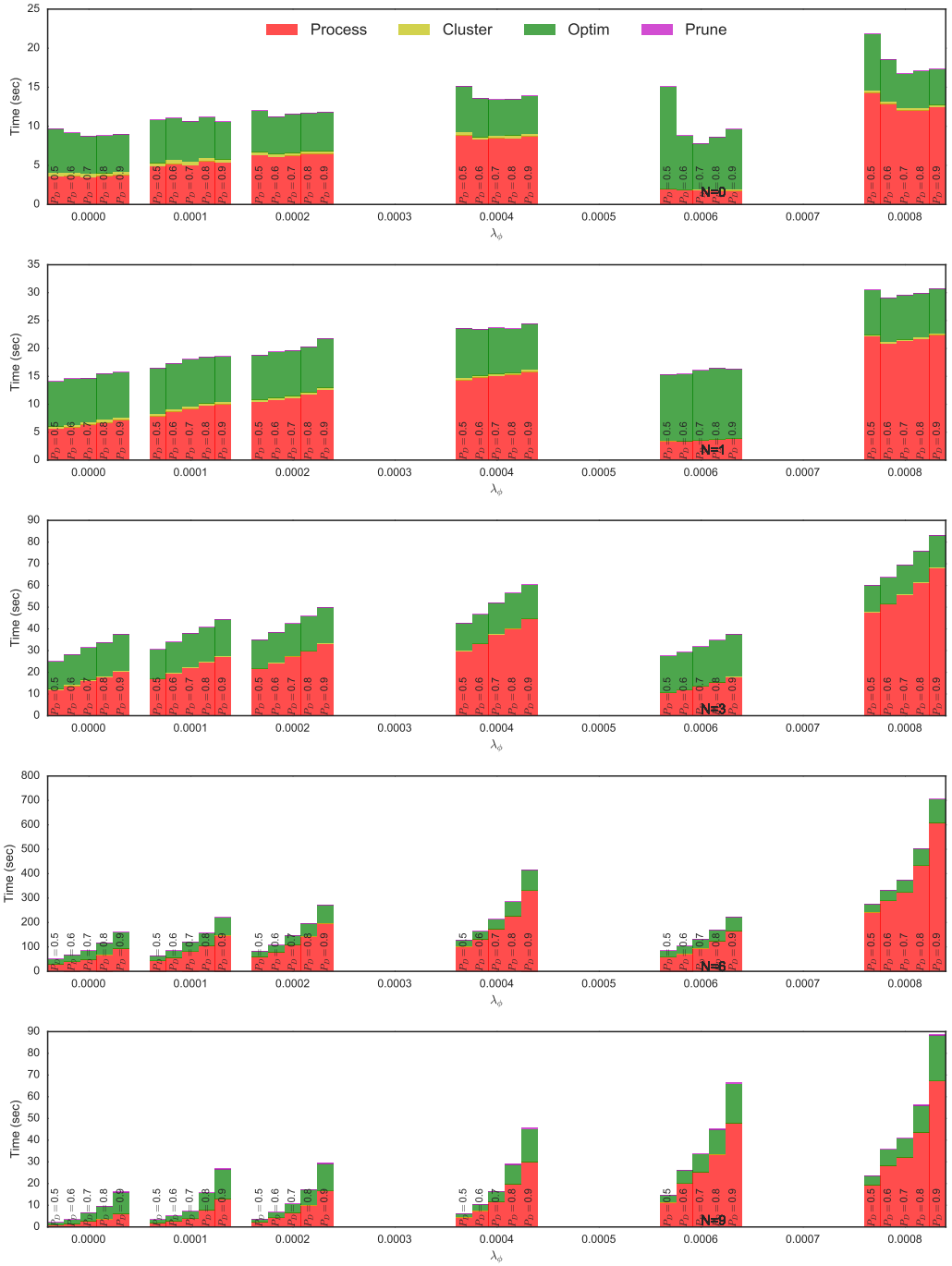


Figure 42: Scenario 6 - Run time log, GLPK solver

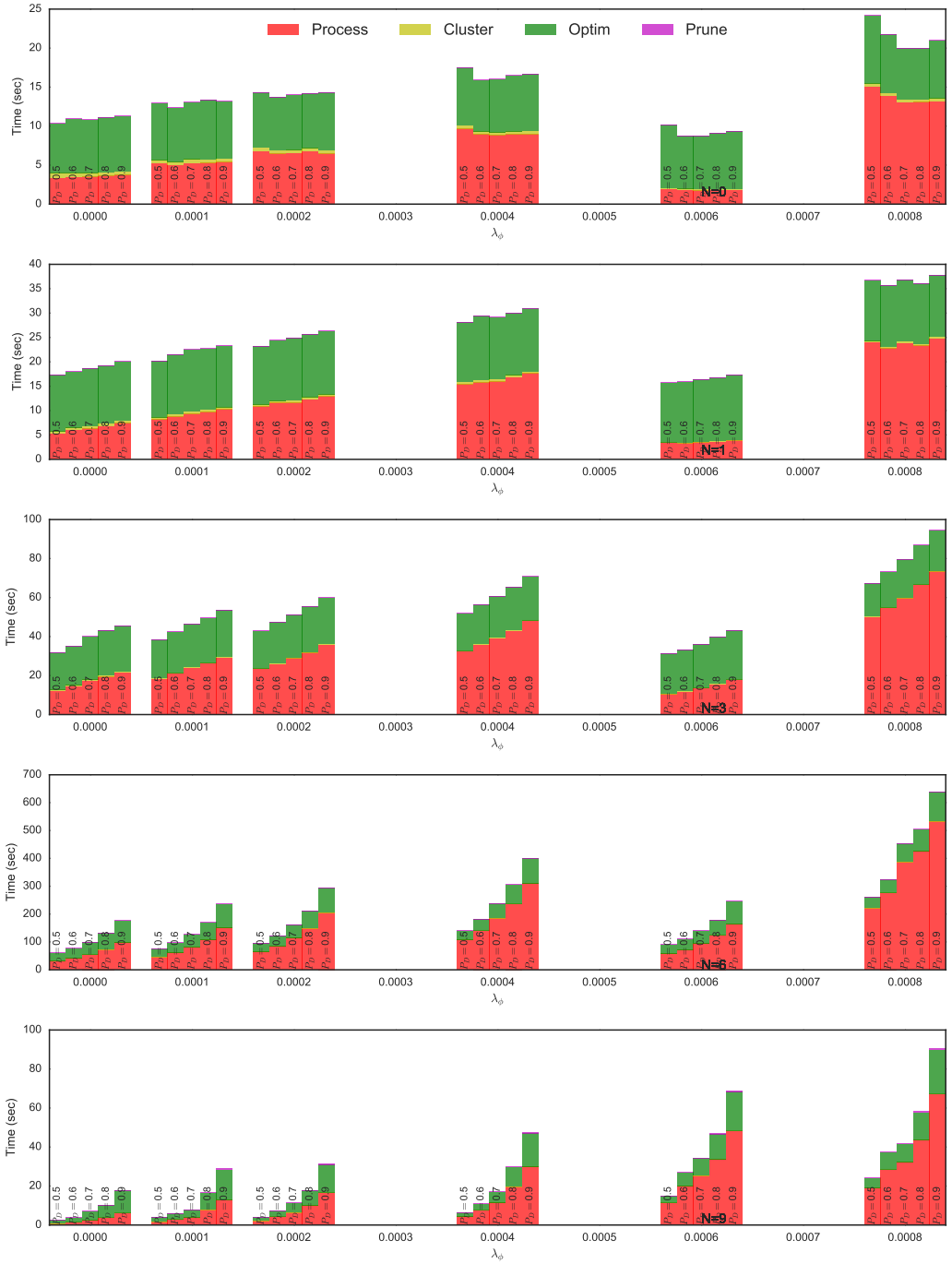


Figure 43: Scenario 6 - Run time log, GUROBI solver

References

[1] Y. Bar-Shalom and T. E. Fortmann, "Algorithms for Tracking A Single Target In Clutter," in *Tracking and Data Association*, pp. 119–185, New York: Academic, 1998.

[2] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Sonar tracking of multiple targets using joint probabilistic data association," *IEEE Journal of Oceanic Engineering*, vol. 8, no. 3, pp. 173–184, 1983.

[3] H. A. P. Blom and E. Bloem, "Probabilistic Data Association Avoiding Track Coalescence," in *IEEE Transactions on Automatic Control (2000)*, vol. 45, pp. 247–259, 200.

[4] QinetiQ, "Efficient hypothesis management," 2003.

[5] P. Horridge and S. Maskell, "Real-time tracking of hundreds of targets with efficient exact JPDAF implementation," in *2006 9th International Conference on Information Fusion, FUSION*, 2006.

[6] R. Singer, R. Sea, and K. Housewright, "Derivation and evaluation of improved tracking filter for use in dense multitarget environments," *IEEE Transactions on Information Theory*, vol. 20, no. 4, pp. 423–432, 1974.

[7] D. B. Reid, "An algorithm for tracking multiple targets," *1978 IEEE Conference on Decision and Control including the 17th Symposium on Adaptive Processes*, vol. 17, no. 6, pp. 843–854, 1978.

[8] T. Kurien, "Issues in the design of practical multitarget tracking algorithms," in *Multitarget-Multisensor Tracking: Application and Advances*, ch. 3, pp. 219–245, Artech House, first ed., 1990.

[9] Y. Bar-Shalom, S. S. Blackman, and R. J. Fitzgerald, "Dimensionless score function for multiple hypothesis tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 1, pp. 392–400, 2007.

[10] S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 5–18, 2004.

[11] S. S. Blackman, R. J. Dempster, and R. W. Reed, "Demonstration of Multiple Hypothesis Tracking (MHT) Practical Real-Time Implementation Feasibility," *Proceedings of SPIE Vol. 4473*, vol. 4473, pp. 470–475, 2001.

[12] P. Storms and F. Spieksma, "An LP-based algorithm for the data association problem in multitarget tracking," *Computers and Operations Research*, vol. 30, no. 7, pp. 1067–1085, 2003.

[13] S. Coraluppi, C. Carthel, M. Luettgen, and S. Lynch, "All-Source Track and Identity Fusion," no. June, 2000.

[14] E. Brekke, O. Hallingstad, and J. Glattetre, "Improved target tracking in the presence of wakes," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 2, pp. 1005–1017, 2012.

[15] C. L. Morefield, "Application of 0–1 Integer Programming to Multitarget Tracking Problems," *IEEE Transactions on Automatic Control*, vol. AC-22:302, pp. 303–312, 1977.

[16] S. Coraluppi and C. Carthel, "Multi-hypothesis sonar tracking," *7 th International Conference on Information Fusion.*, pp. 33–40, 2004.