An extended framework for the analysis of deprioritised algorithms

Stephen Howe
Centre for Mathematics and its Applications
Mathematical Sciences Institute
The Australian National University
Stephen.Howe@anu.edu.au

Abstract

Many important optimisation problems are too difficult to solve exactly. However reasonable approximations can be obtained using randomised heuristics. For example, deprioritised algorithms have been successfully applied to many NP-hard optimisation problems for random regular graphs. Since directed graphs arise naturally in many areas of mathematics and computer science, we would like to analyse deprioritised algorithms designed for directed graphs. Wormald has provided a general method for analysing many deprioritised algorithms, but this method is not applicable to all algorithms for directed graphs. So we extend Wormald’s theory to cover a larger class of deprioritised algorithms. The new results are demonstrated by analysing an algorithm for random $d$-in $d$-out directed graphs which finds a certain type of dominating set.

1 Introduction

Deprioritised algorithms, introduced by Wormald [13], have been used to obtain bounds for many NP-hard optimisation problems on random regular graphs [1, 2, 3, 4]. In particular, asymptotically almost sure (a.a.s.) bounds are obtained, that is, bounds which hold with probability tending to one. Bounds on NP-hard optimisation problems for directed graphs are also of interest. So we would like to analyse deprioritised algorithms defined for random $d$-in $d$-out directed graphs (where $d$-in $d$-out means every vertex has in-degree $d$ and out-degree $d$). For example, a deprioritised algorithm for small dominating sets of random 2-in 2-out directed graphs [6] has been considered.

Wormald [13] provided a framework for analysing many deprioritised algorithms. However many algorithms cannot be analysed using this theory, including many natural algorithms on directed $d$-in $d$-out graphs. We consider one such algorithm, called PathDomSetD, in this paper. So we extend Wormald’s theory to a larger class of deprioritised algorithms. We then demonstrate our results by analysing PathDomSetD. We choose to analyse PathDomSetD because it is simple and yet demonstrates the new features of the theory presented. More significant applications are given in a subsequent
paper [5]. The results we present form the basis of the author’s PhD thesis [7], in which more detail can be found.

A deprioritised algorithm is defined using a given prioritised algorithm. Prioritised algorithms proceed via a sequence of operations where each operation has one of a finite number of types. At each step of the algorithm the type of the next operation performed is chosen using a prioritisation of the types. Deprioritised algorithms also proceed operation by operation. However, deprioritised algorithms choose the type of the next operation randomly. In particular, deprioritised algorithms specify a probability distribution, which changes during the execution of the algorithm, that is used to select the type of the next operation.

We analyse the asymptotic behaviour of a deprioritised algorithm by defining random variables $Y_i(t)$ on the random process determined by the algorithm. Under certain conditions, the scaled random variables $y_i(t/n) = Y_i(t)/n$ are closely approximated (as $n \to \infty$) by the solution to differential equations. These differential equations depend on what is called the phase of the algorithm. We obtain the differential equations for a given phase by considering the expected change in the random variables due to an operation of that phase.

As mentioned above, Wormald [13] has given a general method to analyse some deprioritised algorithms. Wormald’s analysis applies to deprioritised algorithms for which (i) only two types of operations are performed in any given phase (the types depend on the phase), and (ii) each phase (which are also classified by types) has a distinct type. Many algorithms do not satisfy these assumptions.

Our main contribution is to extend Wormald’s theory so that a larger class of deprioritised algorithms may be analysed. This class includes the algorithm PathDomSetD, which cannot be analysed by Wormald’s theory. PathDomSetD is based on a prioritised algorithm PathDomSetP, defined in Section 2, which finds a certain type of dominating set of a random $d$-in $d$-out directed graph. Our theory applies to deprioritised algorithms defined using any prioritised algorithm which satisfies the properties given at the start of Section 3. This definition allows us to define deprioritised algorithms with phases consisting of operations of any number of types and with any sequence of phase types. The differential equations that, under certain conditions, approximate the scaled random variables $y_i(t/n) = Y_i(t)/n$ are determined in Section 3.2. We finish Section 3 by presenting pseudo-code for the deprioritised algorithm based on a given prioritised algorithm.

The analysis of the deprioritised algorithms defined in Section 3 is given in Section 4. The major result of this paper is Theorem 4.8, which states that, when certain hypotheses are satisfied, the behaviour of the deprioritised algorithm is described by the solutions to the differential equations obtained in Section 3.2. The hypotheses are developed at the beginning of Section 4. A number of other results are given (including alternative hypotheses) which make the analysis easier. Finally, in Section 5, we demonstrate the results of Section 4 by analysing PathDomSetD.

2 A Prioritised Algorithm

Throughout this paper we consider a simple, example algorithm PathDomSetP, which we introduce in this section. The algorithm was studied as part of the authors PhD
thesis [7, Section 6.2]. We start with some basic definitions.

A simple directed graph or digraph $G$ is a set $V = V(G)$ of vertices and a set $E = E(G)$ of (directed) edges where each edge is an element of $\{(u, v) \in V^2 : u \neq v\}$. We also consider multi-diagraphs where $E$ is a multi-set of elements from $V^2$. By digraph we always mean a simple digraph and we always take $V = \{1, \ldots, n\}$. The definitions that follow hold for both simple and multi-diagraphs.

For a vertex $u \in V$, vertices in the set $N_{in}(v) = \{v \in V : (v, u) \in E\}$ are the in-neighbours of $u$, while vertices in the set $N_{out}(v) = \{v \in V : (u, v) \in E\}$ are the out-neighbours of $u$. We then define the in-degree of $u$ to be $|N_{in}(u)|$ and the out-degree of $u$ to be $|N_{out}(u)|$. We denote the in-degree and the out-degree of $u$ by $\text{in-deg}(u)$ and $\text{out-deg}(u)$ respectively. The pair whose first component is the in-degree of $u$ and whose second component is the out-degree of $u$ is called the degree pair of $u$, which we denote by $\text{deg-pair}(u)$. For a fixed positive integer $d$, a digraph $G$ is $d$-in $d$-out or regular (of degree $d$) if every vertex of $G$ has degree pair $(d, d)$. We always take $d$ to be fixed.

Now, a dominating set of a digraph $G$ is a subset $D$ of the vertices $V(G)$ such that, for each vertex $u \notin D$, there is an edge from some vertex in $D$ to $u$. If instead, for each vertex $u \notin D$, there is an edge from $u$ to some vertex in $D$, then $D$ is an absorbent set. We will call a set that is both dominating and absorbent a path dominating set [7, Section 2.2]. The algorithm PathDomSetP will find a path dominating set. We are most interested in path dominating sets of minimum size. We denote the minimum size of a path dominating set of $G$ by $\gamma_{\text{path}}(G)$.

We use PathDomSetP to determine an a.a.s. upper bound on $\gamma_{\text{path}}(G)$ for $G \in \mathcal{DG}_{n,d}$ where $\mathcal{DG}_{n,d}$ is the uniform model of $d$-in $d$-out digraphs (on the vertex set $\{1, \ldots, n\}$). Since we are working with random digraphs, the results we obtain will involve probabilities. We often randomly select an element from a given set using the uniform distribution. When we make such a selection, we say that we are selecting uniformly at random (u.a.r.). We denote the probability of an event $A$ by $\mathbb{P}(A)$ and the expected value of a random variable by $\mathbb{E}(\cdot)$. Note that each event and random variable is indexed by $n$, though usually we do not make this explicit. Now consider a property $P$ of $d$-in $d$-out digraphs (for example). Let $A_n$ be the event that $G \in \mathcal{DG}_{n,d}$ has property $P$. Then we say that $P$ holds a.a.s. in $\mathcal{DG}_{n,d}$ if $\mathbb{P}(A_n) \rightarrow 1$ as $n \rightarrow \infty$.

The basic strategy of PathDomSetP is as follows. Starting with an empty digraph, PathDomSetP adds edges randomly until each vertex has degree pair $(d, d)$. In particular, the edges are added using the pairing process described below. As the edges are chosen, a path dominating set $P$ for the constructed digraph $G$ is identified. Analysing PathDomSetP we are able to determine $|P|$ a.a.s. from which we obtain an upper bound on $\gamma_{\text{path}}(G)$ for $G \in \mathcal{DG}_{n,d}$.

### 2.1 Pairing models

The pairing model for $\mathcal{DG}_{n,d}$ has previously appeared [9, 10] with varying terminology. In this model, we start with two sets of $nd$ points; points of one set are called in-points and points of the other set are called out-points. With each vertex in the set $\{1, \ldots, n\}$, we associate $d$ in-points and $d$ out-points so that each point is associated with exactly one vertex. A pairing is then a partition of the set of points (both in and out) into $nd$ blocks of size 2 such that each block contains exactly one in-point and exactly one out-point. Each block of the pairing is called a pair.
For each pairing $P$ there is a corresponding $d$-in $d$-out multi-digraph $DG(P)$. In particular, $DG(P)$ is the multi-digraph such that, for every pair $\{p_{in}, p_{out}\}$ of $P$ (where $p_{in}$ is the in-point and $p_{out}$ is the out-point), $DG(P)$ contains an edge from the vertex associated with $p_{out}$ to the vertex associated with $p_{in}$. Let $\mathcal{DP}_{n,d}$ denote the uniform model of a random pairing for a $d$-in $d$-out digraph on the vertices $\{1, \ldots, n\}$. Then for $P \in \mathcal{DP}_{n,d}$, the digraph $DG(P)$ is a random $d$-in $d$-out multi-digraph distributed non-uniformly. Conditioning on $DG(P)$ having no loops and no multiple edges we obtain the uniform model $\mathcal{DG}_{n,d}$. Moreover, any property that holds a.a.s. for $DG(P)$ with $P \in \mathcal{DP}_{n,d}$, also holds a.a.s. for random $d$-in $d$-out digraphs.

The algorithm PathDomSetP proceeds by constructing a pairing using the pairing process. The pairing process starts with the empty set $P_0$. At each step $t$, we obtain $P_{t+1}$ by adding a pair $\{p_{in}, p_{out}\}$ to $P_t$ such that either $p_{in}$ is chosen u.a.r. from the set of free in-points or $p_{out}$ is chosen u.a.r. from the set of free out-points. The process ends at step $F$ when $P_F$ is a pairing (which is distributed uniformly). During the process, if a point does not occur in a pair of $P_t$ we say the point is free; otherwise we say the point has been exposed. Adding the pair $\{p_{in}, p_{out}\}$ to $P_t$ is called exposing $p_{in}$ (or $p_{out}$, or the pair $\{p_{in}, p_{out}\}$, or the edge to which $\{p_{in}, p_{out}\}$ corresponds).

An algorithm using this pairing process defines a random sequence of sets of pairs $P_0, \ldots, P_F$, which in turn define a random multi-digraph process $G_0, \ldots, G_F$. We define $V(i,j)(t) = V(i,j)(G_t)$ to be the set of vertices of degree pair $(i,j)$ in $G_t$, and further let $Z(i,j)(t) = |V(i,j)(t)|$. We also define other random variables of interest, for example a random variable counting the number of vertices added to the path dominating set.

2.2 The algorithm PathDomSetP

PathDomSetP starts with an empty set of pairs $P$ and an empty set of vertices $D$, and executes via a sequence of operations. There are $2d + 1$ types of operations, each corresponding to a degree pair $(p,q)$ with $p = 0$ or $q = 0$. An operation of type corresponding to the degree pair $(p,q)$ proceeds as follows:

(i) a vertex $u$ of degree pair $(p,q)$ is selected uniformly at random;

(ii) if $p = 0$, then

(a) a free in-point associated with $u$ is exposed to obtain an in-neighbour $w_1$,

(b) the vertex $w_1$ is added to $D$, and

(c) the free points associated with $w_1$ are exposed;
(iii) if \( q = 0 \), then

(a) a free out-point associated with \( u \) is exposed to obtain an out-neighbour \( w_2 \),
(b) the vertex \( w_2 \) is added to \( D \), and
(c) the free points associated with \( w_2 \) are exposed.

We say the operation processes the vertex \( u \). Now the type of an operation processing a vertex of degree pair \((p,q)\) is \(2(p + q) - \delta_{p>q} \) where \( \delta_{p>q} = 1 \) if \( p > q \) and \( \delta_{p>q} = 0 \) if \( p \leq q \). PathDomSetP performs operations while there are vertices with in-degree 0 or out-degree 0. The type \( \tau \) of the next operation performed is the largest type of an operation that can be performed, that is, 
\[
\tau = \max \{2(p + q) - \delta_{p>q} : Y_{(p,q)} > 0 \}.
\]
When there are no vertices with in-degree zero or out-degree zero, PathDomSetP performs no further operations and adds pairs to \( P \) u.a.r. until \( P \) is a pairing.

Notice that, after each operation, every edge exposed has at least one incident vertex in the set \( D \). Operations are performed until every vertex has at least one in-neighbour and at least one out-neighbour. Therefore, when PathDomSetP finishes, the set \( D \) is a path dominating set for \( DG(P) \).

### 3 Prioritised and Deprioritised Algorithms

We now introduce a general definition of prioritised algorithms. Let \( P \) be a randomised algorithm defined on sets \( \Omega_n \) for \( n \geq 1 \). For each \( n \), we assume that for any element \( G_0(n) \in \Omega_n \) the algorithm \( P \), given \( G_0(n) \) as input, generates a random process \( \{G_t\}_{i=0}^{F(n)} \) on \( \Omega_n \). For example, for PathDomSetP, the set \( \Omega_n \) is the set of all multi-digraphs on the vertex set \( \{1, \ldots, n\} \) for which each vertex has in-degree at most \( d \) and out-degree at most \( d \), and \( G_0(n) \) is the empty digraph. We define and analyse a deprioritised algorithm based on \( P \) if \( P \) satisfies the following properties.

**Properties 3.1** (Prioritised algorithms).

(i) The random process \( \{G_t\}_{i=0}^{F} \) on \( \Omega_n \) is Markovian.

(ii) The algorithm proceeds via a sequence of operations \( \text{op}_1, \ldots, \text{op}_F \) such that \( G_t \) is obtained from \( G_{t-1} \) via \( \text{op}_t \) for \( t = 1, \ldots, F \). Moreover, the operations are classified into types. That is, for some fixed integer \( k \), operation \( \text{op}_t \) has type \( \tau(\text{op}_t) \in \{0, 1, \ldots, k\} \), for \( t = 1, \ldots, F \).

(iii) There exists random variables \( Y_0, \ldots, Y_m \) defined on \( \Omega_n \) (for all \( n \)) such that, for \( i = 0, \ldots, m \) and \( r = 0, \ldots, k \), we have
\[
\mathbb{E}(Y_i(G_{t+1}) - Y_i(G_t) | G_t, \tau(\text{op}_t) = r) = f_{i,r}(t/n, Y_0(G_t)/n, \ldots, Y_m(G_t)/n) + o(1) \tag{3.1}
\]
for some functions \( f_{i,r} \). Moreover, for some polynomial \( H \), each function \( f_{i,r} \) has the form
\[
f_{i,r}(x, y_0, \ldots, y_m) = \frac{g_{i,r}(x, y_0, \ldots, y_m)}{(H(x, y_0, \ldots, y_m))^{\ell_i}} \tag{3.2}
\]
for some integer \( \ell_i \geq 1 \) and some polynomial \( g_{i,r} \).
(iv) For \( r = 0, \ldots, k \), an operation of type \( r \) can be performed on every \( G \in \Omega_n \) for which \( Y_r(G) > 0 \). When \( Y_r(G) > 0 \) we say that an operation of type \( r \) is permissible (on \( G \)). Then, given \( G_t \), the type \( \tau(\text{op}_{t+1}) \) of the next operation is the maximum of all types which are permissible on \( G_t \).

(v) For some fixed \( \beta \) we have \( \max_{0 \leq t \leq m} |Y_i(G_{t+1}) - Y_i(G_t)| \leq \beta \) for all \( t \geq 0 \).

(vi) For some fixed \( M \) we have \( |Y_i(G_t)| \leq Mn \) for \( i = 0, \ldots, m \) and all \( t \geq 0 \).

(vii) The limits \( \lim_{n \to \infty} Y_i(0)/n \) exist for \( i = 0, \ldots, m \).

Parts (i)–(iv), excluding the restriction on the form of the functions \( f_{i,r} \), are fundamental to prioritised algorithms. The remaining parts are restrictions that allow the analysis presented in Section 4. The required form of the functions \( f_{i,r} \) is reasonable since we consider combinatorial algorithms. Also part (v) could be relaxed [12, Theorem 5.1]. Since we are interested in the behaviour of algorithms as \( n \) tends to infinity, we often drop \( n \) from the notation.

Throughout this paper, we consider two random processes on \( \Omega_n \). One defined by a prioritised algorithm and one defined by a deprioritised algorithm. The random variables \( Y_i \) are defined on both random processes. So we let \( Y_i(t) = Y_i(G_t) \) for the random process \( \{G_t\}_{t=0}^{\infty} \) when the random process being considered is clear from the context. Notice that the functions \( f_{i,r} \) depend only on the random variables. So the functions \( f_{i,r} \) are also defined for the deprioritised algorithm. We let

\[
 f_{i,r}(t) = f_{i,r}(t/n, Y_0(t)/n, \ldots, Y_m(t)/n)
\]

when the underlying random process is clear from the context. Later we consider continuous functions corresponding to the scaled random variables. So we also let

\[
 f_{i,r}(x) = f_{i,r}(x, y_0(x), \ldots, y_m(x))
\]

when the functions \( y_i \) are clear from the context.

At this point, it is useful to introduce the domain

\[
 D(\delta) = \{(x, y_0, \ldots, y_m) : -\delta < x < C, \ y_0 > \delta, \ H > \delta, \ |y_i| < 2M \ (i = 0, \ldots, m)\} \tag{3.3}
\]

for \( \delta > 0 \). Later, in Section 3.2, we use \( D(\delta) \) to define functions which approximate the scaled random variables of the deprioritised algorithm. Notice that the condition \( H > \delta \) ensures that the functions \( f_{i,r} \) are defined on \( D(\delta) \). We will discuss the domain \( D(\delta) \) further in Section 3.2.

In Wormald’s analysis [13, Theorem 2] of deprioritised algorithms, the form of the functions \( f_{i,r} \) is also used. In particular, Wormald requires that these functions satisfy

\[
 C_1 y_{i+1} - C_2 y_i \leq f_{i,r} \leq C_3 y_{i+1} \tag{3.4}
\]

for positive constants \( C_1 \), \( C_2 \), and \( C_3 \) on an appropriate domain. However, as we shall see in Section 5, the functions \( f_{i,r} \) for \text{PathDomSetP} have the form

\[
 f_{i,r} = \delta_{i,r} - py_i + q_1 y_{j_1} + q_2 y_{j_2}
\]

where \( p \), \( q_1 \), and \( q_2 \) are polynomials, such that \( y_{j_2} \) does not divide \( q_1 \) and \( y_{j_1} \) does not divide \( q_2 \). Thus there is no variable \( y_j \) and constant \( C \) for which \( f_{i,r} \leq Cy_j \) and so we cannot apply the earlier work of Wormald.
3.1 Deprioritised algorithms

Deprioritised algorithms are based on prioritised algorithms. So the deprioritised algorithm uses the same types of operations as the prioritised algorithm. However, deprioritised algorithms select the type of the next operation according to a probability distribution. We call this distribution the type distribution.

Recall that a deprioritised algorithm goes through a sequence of phases. To ensure that we can perform an operation of the type selected, we start each phase of the deprioritised algorithm with a preprocessing subphase. The remainder of the phase is called the main subphase. In the preprocessing subphase only operations of type 0 are performed. This should allow (a.a.s.) the other operation types to become permissible. As long as the preprocessing subphase is sufficiently short, its impact on the behaviour of the algorithm is negligible.

For a main subphase, we choose the type distribution so that the probability that the next operation has type $r$ to approximate the proportion of type $r$ operations at a similar stage of the prioritised algorithm (by a similar stage, we mean when the values of the random variables are similar). Unfortunately, it is not possible to be precise about the proportion of type $r$ operations in the prioritised algorithm. However, we still define a type distribution with this goal in mind.

3.1.1 Type distributions

We start by considering the operations of a prioritised algorithm. Let $O = \{w_1, \ldots, w_a\}$ be a subset of $\{0, \ldots, k\}$ such that $w_1 < w_2 < \cdots < w_a$. Then an $O$-clutch is a subsequence of operations of a prioritised algorithm consisting of an initial operation of type $w_1$, followed by each subsequent operation of a type from $O \setminus \{w_1\}$, up to (but not including) the next operation of a type not in $O \setminus \{w_1\}$. Note that we can also define $O$-clutches recursively. Let $O_i = O \setminus \{w_1, \ldots, w_{i-1}\} = \{w_i, \ldots, w_a\}$. Then an $O$-clutch is an operation of type $w_1$, followed by zero or more $O_a$-clutches, followed by zero or more $O_{a-1}$-clutches, and so on, until ending with zero or more $O_2$-clutches.

Now fix a particular $O$-clutch $\Gamma$ occurring near step $t$. Let $n_r$ be the expected number of type $r$ operations in $\Gamma$. For $j \geq 1$, the random variable $Y_{w_j}$ is zero at the beginning and end of $\Gamma$. So the expected change in $Y_{w_j}$ due to $\Gamma$ is $\sum_{i=1}^{a} p_{w_i, w_j, w_i} = 0$. We also have $n_{w_1} = 1$. Note that these equations have previously been considered by Shi and Wormald [11]. Thus the proportion $p_r = n_r / \sum_{r=0}^{a} n_r$ of operations of type $r$ satisfies

$$\sum_{i=1}^{a} p_{w_i, w_j, w_i} = 0 \text{ (for } j = 2, \ldots, a) \quad \text{and} \quad \sum_{i=1}^{a} p_{w_i} = 1.$$

We can write the above equations as the matrix equation $(p_{w_1}, \ldots, p_{w_a}) C = (1, 0, \ldots, 0)$, where $C$ is called the clutch matrix and is defined in Definition 3.2 below. Applying Cramer’s rule we find an explicit formula for $p_{w_i}$. Note that solving the matrix equation numerically is a more efficient way of determining the values of $p_{w_i}$ than using the explicit formula in Definition 3.2. However the explicit formula is useful for determining properties of the functions $p_r$ and functions defined in terms of $p_r$. We determine such properties later in this section and in Section 4.

In Definition 3.2 we also define functions $\Phi_b$. We believe that $\Phi_b$ approximates the expected change in $Y_{w_b}$ due to an $O_b$-clutch. So if $\Phi_b(t) < 0$ for $b = 2, \ldots, a$, then $Y_{w_b}$
should decrease between type $w_b$ operations and an $\mathcal{O}$-clutch should be short. Thus the operations around operation $t$ can be grouped into many short $\mathcal{O}$-clutches. This is not the case when $\Phi_b(t) > 0$ for some $b \geq 2$, since then there are no operations of types less than $w_b$. So, at the point when $\Phi_b$ becomes positive for some $b \geq 2$, the behaviour of the algorithm changes. Thus we use the functions $\Phi_b$ to define the phases of the deprioritised algorithm.

The functions $p_r$ and $\Phi_b$ are defined using determinants of submatrices of the clutch matrix. For submatrices of an arbitrary matrix we use the following notation: for an $m \times m$ matrix $A$ and sets $R, C \subseteq \{1, \ldots, m\}$ we denote by $A(R, C)$ the matrix obtained from $A$ by deleting the rows and columns with indices in $R$ and $C$ respectively. If $R = \{r\}$ and $C = \{c\}$, then we write $A(R, C)$ as $A(r, c)$. It is also useful to define the determinant of the matrix with zero rows and zero columns to be 1.

**Definition 3.2** (The Proportions of Operation Types). Let $\mathcal{O} = \{w_1, \ldots, w_a\}$ with $w_1 < \cdots < w_a$. The **clutch matrix** for an $\mathcal{O}$-clutch at step $t$ is

$$
\mathcal{C}(t) = \begin{pmatrix}
1 & f_{w_2,w_1}(t) & \cdots & f_{w_a,w_1}(t) \\
& \vdots & & \vdots \\
1 & f_{w_2,w_a}(t) & \cdots & f_{w_2,w_a}(t)
\end{pmatrix}.
$$

Then, for $b = 2, \ldots, |\mathcal{O}|$, we define

$$
\Phi_b(t) = \frac{\det \mathcal{C}(t)(\{1, \ldots, b-1\}, \{1, \ldots, b-1\})}{\det \mathcal{C}(t)(\{1, \ldots, b\}, \{1, \ldots, b\})},
$$

and, for $r = 0, \ldots, k$, we define

$$
p_r(t) = \begin{cases}
0 & \text{if } r \notin \mathcal{O}, \\
(-1)^{b+1} \frac{\det \mathcal{C}(t)(b,1)}{\det \mathcal{C}(t)} & \text{if } r = w_b \in \mathcal{O}.
\end{cases}
$$

Notice that the clutch matrix and the functions $p_r$ and $\Phi_b$ depend only on $\mathcal{O}$ and the random variables $Y_i$ (via the functions $f_i,r$). So $\mathcal{C}(t)$, $\Phi_b(t)$, and $p_r(t)$ are defined for both a prioritised algorithm and the corresponding deprioritised algorithm.

We may also derive $\mathcal{C}(t)$, $\Phi_b(t)$, and $p_r(t)$ by considering an $\mathcal{O}$-clutch $\Gamma$ to be an $(\mathcal{O}\{w_2\})$-clutch followed by a sequence of $\mathcal{O}_2$-clutches. Then $\Gamma$ is similar to a clutch of two operations, where instead of operations we have $\mathcal{O}\{w_2\}$ and $\mathcal{O}_2$ clutches. We may then obtain Definition 3.2 by induction [7, Section 4.2.1].

We use Definition 3.2 together with sets of types called **irreducible type sets** to define the type distributions for the main subphases of a deprioritised algorithm. Note that the type distribution depends on the phase of the algorithm. In particular, we classify phases by types and the type distribution depends on the type of the phase. Irreducible type sets are determined by the prioritised algorithm and are defined later.

**Definition 3.3** (Type Distributions). Let $\mathcal{P}$ be an algorithm satisfying Properties 3.1 and let $\mathcal{M}^{(\tau)}$ be the irreducible type set for a phase of type $\tau$ of $\mathcal{P}$. Then the type distribution for a phase of type $\tau$ of the deprioritised algorithm based on $\mathcal{P}$ is defined by the functions $p_r^{(\tau)}(t) = p_r(t)$ (for $r = 0, \ldots, k$) obtained via Definition 3.2 with $\mathcal{O} = \mathcal{M}^{(\tau)}$. In particular, the probability that operation $t$, occurring during a phase of type $\tau$, has type $r$ is $p_r^{(\tau)}(t)$.  

$\diamond$
We also define $C^{(r)}(t)$ to be the clutch matrix of an $\mathcal{M}^{(r)}$-clutch at step $t$ and let $\Phi^{(r)}_b(t) = \Phi_b(t)$ via Definition 3.2 with $\mathcal{O} = \mathcal{M}^{(r)}$.

Note that the functions $p_r^{(r)}(t)$ (as a probability distribution) must satisfy $p_r^{(r)}(t) \geq 0$ (for $r = 0, \ldots, k$) and $\sum_{r=0}^{k} p_r^{(r)}(t) = 1$. The later condition follows from the form of the functions (see (3.6)). We ensure that each function is non-negative by restricting the definition of a phase of type $\tau$. For technical reasons we also require that $p_r^{(r)}(t)$ be positive for $r \in \mathcal{M}^{(r)}$. The functions $p_r^{(r)}(t)$ must also be Lipschitz as they are used to define differential equations. So we also require that $|\det C^{(r)}(t)| > \delta$ for some constant $\delta > 0$. Thus for a deprioritised algorithm to be in a phase of type $\tau$ at step $t$, we require that (a) $|\det C^{(r)}(t)| > \delta$ for some $\delta > 0$, (b) $p_r^{(r)}(t) > 0$ for $r \in \mathcal{M}^{(r)}$, and (c) $\Phi_b^{(r)}(t) < 0$ for $b = 2, \ldots, |\mathcal{M}^{(r)}|$. The following conditions are equivalent to (a), (b), and (c). As they are easier to check, we use them when analysing each main subphase of a deprioritised algorithm.

**Lemma 3.4** ([7, Lemma 4.2.3]). Let $a = |\mathcal{M}^{(r)}|$. For all $t$, the conditions (a), (b), and (c) above are equivalent to the following:

(i) $(-1)^{a+1} \det C^{(r)}(t) > \delta$ for some $\delta > 0$,

(ii) $(-1)^{a+b} \det C^{(r)}(t)(b, 1) > 0$ for $b = 1, \ldots, a$, and

(iii) $(-1)^{a-1} \det C^{(r)}(t)(\{1, \ldots, b-1\}, \{1, \ldots, b-1\}) > 0$ for $b = 2, \ldots, a$.

Also, conditions (i) and (ii) imply (a) and (b).

### 3.1.2 Irreducible type sets

The irreducible type sets are defined in terms of a digraph called the expected change digraph. Recall the domain $D(\delta)$ defined by (3.3).

**Definition 3.5** (Expected Change Digraph (ECD)). The expected change digraph (or ECD) for a phase of type $\tau$ is the digraph $\Gamma(\tau)$ on the vertices $\{\tau, \ldots, k\}$ for which there is an edge from $w_1$ to $w_2$ if and only if $w_1 \neq w_2$ and there exists $(x, y_0, \ldots, y_m) \in D(\delta)$ such that $y_{r+1} = \cdots = y_k = 0$ and $f_{w_2, w_1}(x, y_0, \ldots, y_m) \neq 0$.

To check the condition above, we can substitute $y_i = 0$ for $i = \tau + 1, \ldots, k$ in $f_{w_2, w_1}$ and simplify the resulting expression. We will obtain either 0 or a polynomial in the variables $x, y_0, \ldots, y_{\tau+1}, \ldots, y_m$.

**Definition 3.6** (Irreducible Type Set). The irreducible type set for a phase of type $\tau$ is the set $\mathcal{M}^{(r)}$ of vertices of $\Gamma(\tau)$ that can be reached from $\tau$ via directed paths.

The most important property of $\mathcal{M}^{(r)}$ is given by the next lemma. It is also useful to note that $\tau \in \mathcal{M}^{(r)}$ always.

**Lemma 3.7.** Fix a type $r \in \{\tau, \ldots, k\}\setminus\mathcal{M}^{(r)}$. Then for all $(x, y_0, \ldots, y_m) \in D(\delta)$ with $y_{r+1} = \cdots = y_k = 0$, we have $f_{r, w}(x, y_0, \ldots, y_m) = 0$ for all $w \in \mathcal{M}^{(r)}$.

**Proof.** Assume for some $r \in \{\tau, \ldots, k\}\setminus\mathcal{M}^{(r)}$ there exists $(x, y_0, \ldots, y_m) \in D(\delta)$ with $y_{r+1} = \cdots = y_k = 0$ and $f_{r, w}(x, y_0, \ldots, y_m) \neq 0$ for some $w \in \mathcal{M}^{(r)}$. Then by definition, there is an edge from $w$ to $r$ in the expected change digraph. Since $w$ lies on a directed path from $\tau$, so too must $r$. Therefore $r \in \mathcal{M}^{(r)}$; a contradiction. \qed
Lemma 3.7 is useful because the functions $\hat{y}_i$ that describe the scaled random variables of the deprioritised algorithm (or so we wish to prove) are such that $\hat{y}_i = 0$ for $i = \tau + 1, \ldots, k$ during a phase of type $\tau$. The functions $f_{i,r}$ often satisfy a stronger property than that implied by Lemma 3.7. When the functions satisfy this property, analysis of the deprioritised algorithm is much easier.

**Definition 3.8** (Independent Types Property). The functions $f_{i,r}$ satisfy the Independent Types Property for a phase of type $\tau$ if, for $r \in \{\tau, \ldots, k\} \setminus M(\tau)$ and for all $(x, y_0, \ldots, y_m) \in D(\delta)$ with $y_{r+1} = \cdots = y_k = 0$, we have $f_{r,w}(x, y_0, \ldots, y_k) = -\delta_{r,w}$ for $w = 0, \ldots, k$. 

During a phase of type $\tau$ of a prioritised algorithm, (we believe that) operations of types from $\{\tau, \ldots, k\} \setminus M(\tau)$ a.a.s. do not occur. Indeed, provided the Independent Types Property holds, using Definition 3.2 with $\mathcal{O} = \{\tau, \ldots, k\}$ and $\mathcal{O} = M(\tau)$ results in the same type distribution. Since we expect most algorithms to satisfy the Independent Types Property for every phase, we choose to define the type distributions using irreducible type sets. This choice should have only a small effect on the subsequent analysis.

### 3.1.3 Determining irreducible type sets

For some algorithms on random $d$-in $d$-out digraphs the irreducible type sets can be determined using the so called Degree Pair Progress Digraph (or DPPD) method. The DPPD method should be easier to apply than using expected change digraphs directly.

**Definition 3.9** (Degree Pair Progress Digraph). Consider an algorithm on a random $d$-in $d$-out digraph satisfying Properties 3.1 and defining the random process $\{G_t\}$. For each degree pair $(i,j)$, let $B_{(i,j)}$ be the minimal set of degree pairs such that, for all $t \geq 1$, if vertex $v$ has degree pair $(i,j)$ in $G_t$, then in $G_{t-1}$ a.a.s. $v$ has degree pair $(p,q) \in B_{(i,j)} \cup \{(i,j)\}$. The degree pair progress digraph is the digraph with the vertex set $\{(i,j) : 0 \leq i, j \leq d\}$ and the edge set $\bigcup_{(i,j)} \{((p,q),(i,j)) : (p,q) \in B_{(i,j)}\}$.

For example, the DPPD for PathDomSetP (with $d = 2$ and vertex $(2,2)$ removed) is shown in Figure 3.1.

To apply the DPPD method, the random variables $Y_0, \ldots, Y_m$ (from Properties 3.1 (iii)) must include the random variables $Z_{(i,j)}$ which count the number of vertices of degree pair $(i,j)$; moreover, each operation type must correspond to a degree pair. So we define a function $\text{dp}: \{0, \ldots, k\} \rightarrow \{(i,j) : 0 \leq i,j \leq d\}$ such that $Y_r = Z_{\text{dp}(r)}$, and a function $\nu:\{(i,j) : 0 \leq i,j \leq d\} \rightarrow \{0,\ldots,m\}$ such that $Z_{(i,j)} = Y_{\nu(i,j)}$. Note that $\nu(\text{dp}(r)) = r$ for $r \in \{0,\ldots,k\}$.

For example, the algorithm PathDomSetP with $d = 2$ has five types of operations. From Section 2 we see that we have $\text{dp}(0) = (0,0)$, $\text{dp}(1) = (1,0)$, $\text{dp}(2) = (0,1)$, $\text{dp}(3) = (2,0)$, and $\text{dp}(4) = (0,2)$. The definition of $\text{dp}$ also partially defines $\nu$; we may complete the definition of $\nu$ in any way.

We determine the irreducible type sets using the degree pair progress digraph as follows. Consider a phase of type $\tau$. We colour black those vertices corresponding (via $\text{dp}$ and $\nu$) to types greater than $\tau$. The remaining vertices, including the vertex $\text{dp}(\tau)$, are coloured white. (Figure 3.2 shows the DPPD for PathDomSetP with $d = 2$ coloured
for a phase of type 1.) Let \( \mathcal{O}(\tau) \) be the subset of \( \{\tau, \ldots, k\} \) such that \( r \in \mathcal{O}(\tau) \) if and only if the vertex \( \text{dp}(r) \) or any of the in-neighbours of \( \text{dp}(r) \) are coloured white. In Lemma 3.10, given below, we prove that under certain conditions the irreducible type set for a phase of type \( \tau \) is \( \mathcal{O}(\tau) \). So for PathDomSetP (with \( d = 2 \)) we have \( \mathcal{O}^{(1)} = \{1, 2, 3\} \); in Section 5 we show that \( \mathcal{O}^{(1)} = \mathcal{M}^{(1)} \) using Lemma 3.10 below.

To apply the DPPD method, properties of the functions \( f_{i,r} \) must be captured by the DPPD. Fix a phase type \( \tau \). Recall that \( f_{i,r} = g_{i,r}/H^{\ell_i} \) for some polynomials \( g_{i,r} \) and \( H \), and some \( \ell_i \geq 1 \). Let \( \Sigma = \{(b, a_0, \ldots, a_m) \in \mathbb{Z}^{m+2} : b, a_0, \ldots, a_m \geq 0 \} \). Then consider the polynomial

\[
g_{i,r} + \delta_{i,r}H^{\ell_i} = \sum_{(b, a_0, \ldots, a_m) \in \Sigma} C(b, a_0, \ldots, a_m)x^by_0^{a_0}\ldots y_m^{a_m}
\]

for real numbers \( C(b, a_0, \ldots, a_m) \). We say that \( f_{i,r} \) is \( B_{\text{dp}(i)} \)-determined if

(a) for all \( (b, a_0, \ldots, a_m) \in \Sigma \) with \( C(b, a_0, \ldots, a_m) \neq 0 \), there exists \( j \in \{i\} \cup \nu(B_{\text{dp}(i)}) \) such that \( a_j > 0 \), and

(b) for \( (p, q) \in B_{\text{dp}(i)} \), there exists \( (b, a_0, \ldots, a_m) \in \Sigma \) with \( a_{\nu(p,q)} > 0 \) and \( a_j = 0 \) for \( j = \tau + 1, \ldots, k \), such that \( C(b, a_0, \ldots, a_m) \neq 0 \).

If \( f_{i,r} \) is \( B_{\text{dp}(i)} \)-determined for all \( i \in \{\tau, \ldots, k\} \) and for all \( r \in \{0, \ldots, k\} \), then the DPPD can be used to determine the irreducible types sets.

**Lemma 3.10.** Define \( \mathcal{O}(\tau) \) as above and recall that \( \mathcal{M}(\tau) \) is the irreducible type set for a phase of type \( \tau \) (see Definition 3.6). If \( f_{i,r} \) is \( B_{\text{dp}(i)} \)-determined for all \( i \in \{\tau, \ldots, k\} \) and for all \( r \in \{0, \ldots, k\} \), then \( \mathcal{O}(\tau) = \mathcal{M}(\tau) \) and the Independent Types Property is satisfied for a phase of type \( \tau \).
Proof. Notice that, in the coloured DPPD (for a phase of type $\tau$), a vertex $(i, j)$ is coloured black if and only if $\nu(i, j) \in \{\tau + 1, \ldots, k\}$. Also $\tau \in \mathcal{O}(\tau)$ and $\tau \in \mathcal{M}(\tau)$ by the definitions of $\mathcal{O}(\tau)$ and $\mathcal{M}(\tau)$.

Consider $w \in \mathcal{O}(\tau)$ with $w \neq \tau$. Then in the coloured DPPD, the vertex $dp(w)$ has at least one white in-neighbour. That is, there exists $(p, q) \in B_{dp(w)}$ such that $\nu(p, q) \notin \{\tau + 1, \ldots, k\}$. Now $f_{w, \tau}$ is $B_{dp(w)}$-determined: so by part (b), setting $y_i = 0$ in $f_{w, \tau}$ for $i = \tau + 1, \ldots, k$ we obtain a non-zero rational function. Hence there exists $(x, y_0, \ldots, y_m) \in D(\delta)$ with $y_{r+1} = \cdots = y_k = 0$ and $f_{w, \tau}(x, y_0, \ldots, y_m) \neq 0$. So in the ECD there is an edge from $\tau$ to $w$ and thus $w \in \mathcal{M}(\tau)$.

Now consider $w \in \{\tau + 1, \ldots, k\} \setminus \mathcal{O}(\tau)$. The vertex $dp(w)$ and its in-neighbours are coloured black. Hence, for all degree pairs $(p, q) \in B_{dp(w)} \cup \{dp(w)\}$, we have that $\nu(p, q) \in \{\tau + 1, \ldots, k\}$. Fix $u \in \{0, \ldots, k\}$. Since $f_{w, u}$ is $B_{dp(w)}$-determined, for all $(x, y_0, \ldots, y_m) \in D(\delta)$ with $y_{r+1} = \cdots = y_k = 0$ we have $f_{w, u}(x, y_0, \ldots, y_m) = -\delta_{w,u}$. Hence, in the ECD, vertex $w$ has zero in-degree and so $w \notin \mathcal{M}(\tau)$. Therefore $\mathcal{O}(\tau) = \mathcal{M}(\tau)$ and, moreover, the Independent Types Property is satisfied for a phase of type $\tau$. 

With a slight adjustment, the DPPD method should be applicable to algorithms for random regular undirected graphs. In this case the corresponding DPPD would be a directed path and the irreducible type sets would have size 1 or 2. Indeed, this is assumed in Wormald’s earlier analysis [13] of deprioritised algorithms. For the algorithms we consider, most irreducible type sets have size greater than two.

3.1.4 Using irreducible type sets

We now justify our choice of defining type distributions using irreducible type sets. The results we present are also used later when analysing deprioritised algorithms.

Let

$$ix(r, S) = \begin{cases} |\{w \in S : w \leq r\}| & \text{if } r \in S, \\ 0 & \text{if } r \notin S. \end{cases}$$

Recall that $p_r^{(\tau)}$ are the functions obtained from (3.6) of Definition 3.2 using $\mathcal{O} = \mathcal{M}(\tau)$.

Lemma 3.11. Let $\mathcal{N} = \{\tau, \ldots, k\}$ and let $n_r$ be the functions (3.6) of Definition 3.2 with $\mathcal{O} = \mathcal{N}$. If the clutch matrix for a $\mathcal{N}$-clutch has non-zero determinant, then the clutch matrix for a $\mathcal{M}(\tau)$-clutch has non-zero determinant, and for all $(x, y_0, \ldots, y_m) \in D(\delta)$ with $y_{r+1} = \cdots = y_k = 0$ we have

$$n_r(x, y_0, \ldots, y_m) = \begin{cases} 0 & \text{if } r \in \mathcal{N} \setminus \mathcal{M}(\tau), \\ p_r^{(\tau)}(x, y_0, \ldots, y_m) & \text{if } r \in \mathcal{M}(\tau). \end{cases}$$

Moreover, if the Independent Types Property for a phase of type $\tau$ is satisfied and the clutch matrix for a $\mathcal{M}(\tau)$-clutch has non-zero determinant, then the clutch matrix for a $\mathcal{N}$-clutch has non-zero determinant.

Proof. Fix $(x, y_0, \ldots, y_m) \in D(\delta)$ with $y_{r+1} = \cdots = y_k = 0$; all functions are evaluated at $(x, y_0, \ldots, y_m)$, although we do not make this explicit. Let $a = |\mathcal{N}|$ and further let $\mathcal{N} \setminus \mathcal{M}(\tau) = \{w_1, \ldots, w_b\}$ where $w_1 < \cdots < w_b$. Note that $\tau \notin \{w_1, \ldots, w_b\}$. Let $\mathcal{D}$ be the clutch matrix for a $\mathcal{N}$-clutch and $\mathcal{C}(\tau)$ be the clutch matrix for a $\mathcal{M}(\tau)$-clutch.
We start by considering \( \mathcal{D} \). Let \( \sigma \) be the permutation of \( \{1, \ldots, a\} \) that maps \( \text{ix}(w_i, \mathcal{N}) \) to \( i \) (for \( i = 1, \ldots, b \)) and maps \( \text{ix}(w, \mathcal{N}) \) to \( \text{ix}(w, \mathcal{M}^{(r)}) + b \) for \( w \in \mathcal{M}^{(r)} \). Then permuting both the rows and columns of \( \mathcal{D} \) by \( \sigma \) we obtain the matrix

\[
\hat{\mathcal{D}} = \begin{pmatrix} A & B \\ O & C \end{pmatrix}
\]

where

\[
A = \begin{pmatrix} f_{w_1, w_1} & \cdots & f_{w_b, w_1} \\ \vdots & \ddots & \vdots \\ f_{w_1, w_b} & \cdots & f_{w_b, w_b} \end{pmatrix},
\]

(3.7)

\( B \) is a \( b \times (a - b) \) matrix, \( O \) is an \( (a - b) \times b \) matrix of zeros (by Lemma 3.7), and

\[
C = \mathcal{D}([\text{ix}(w_1, \mathcal{N}), \ldots, \text{ix}(w_b, \mathcal{N})], \{\text{ix}(w_1, \mathcal{N}), \ldots, \text{ix}(w_b, \mathcal{N})\}) = \mathcal{C}^{(r)}.
\]

Now \( \det \mathcal{D} = \det \hat{\mathcal{D}} = \det A \det \mathcal{C}^{(r)} \). Thus \( \det \mathcal{D} \neq 0 \) implies \( \det \mathcal{C}^{(r)} \neq 0 \). Moreover, when the Independent Types Property is satisfied, we have \( A = \text{diag}(-1, \ldots, -1) \) and so \( \det \mathcal{C}^{(r)} \neq 0 \) implies that \( \det \mathcal{D} \neq 0 \).

Now fix \( r \in \mathcal{N} \setminus \mathcal{M}^{(r)} \). Let \( \mathcal{R} \) be the rows of \( \mathcal{D} \) with indices in \( \{\text{ix}(w, \mathcal{N}) : w \in \mathcal{M}^{(r)}\} \). By Lemma 3.7, each vector in \( \mathcal{R} \) has a zero in columns \( \text{ix}(w_1, \mathcal{N}), \ldots, \text{ix}(w_b, \mathcal{N}) \). Note that these indices are all at least 2 as \( \tau \notin \{w_1, \ldots, w_b\} \). Let \( \hat{\mathcal{R}} \) be the vectors obtained by removing the first component of the vectors of \( \mathcal{R} \). There are \( a - b \) vectors in \( \hat{\mathcal{R}} \) and each vector appears as a row of \( \mathcal{D}(\text{ix}(r, \mathcal{N}), 1) \). So each vector of \( \hat{\mathcal{R}} \) has at least \( b \) zeros, namely in columns \( \text{ix}(w_1, \mathcal{N}) - 1, \ldots, \text{ix}(w_b, \mathcal{N}) - 1 \). Hence the vectors of \( \hat{\mathcal{R}} \) are contained in a subspace of dimension \( a - b - 1 \). So \( \hat{\mathcal{R}} \) is linearly dependent and thus \( \det \mathcal{D}(\text{ix}(r, \mathcal{N}), 1) = 0 \). Therefore

\[
n_r = (-1)^{\text{ix}(r, \mathcal{N})+1} \frac{\det \mathcal{D}(\text{ix}(r, \mathcal{N}), 1)}{\det \mathcal{D}} = 0.
\]

Now fix \( r \in \mathcal{M}^{(r)} \) and let \( i = \text{ix}(r, \mathcal{N}) \). Consider the matrix \( \mathcal{D}(i, 1) \). Let \( \mathcal{E} \) be the matrix obtained from \( \mathcal{D} \) by replacing the entries of row \( i \) and column 1 with zeros, then placing a 1 in the entry with row \( i \) and column 1. Then \( \det \mathcal{E} = (-1)^{i+1} \det \mathcal{D}(i, 1) \). The permutation taking \( \mathcal{D} \) to \( \hat{\mathcal{D}} \) also takes \( \mathcal{E} \) to

\[
\hat{\mathcal{E}} = \begin{pmatrix} A & \hat{B} \\ O & \hat{C} \end{pmatrix}
\]

where \( A \) and \( O \) are as before, \( \hat{B} \) is another \( b \times (a - b) \) matrix, and \( \hat{C} \) is the matrix obtained from \( \mathcal{C}^{(r)} \) by replacing the entries of row \( \text{ix}(r, \mathcal{M}^{(r)}) \) and column 1 with zeros, then placing a 1 in the entry with row \( \text{ix}(r, \mathcal{M}^{(r)}) \) and column 1.

Therefore

\[
(-1)^{i+1} \det \mathcal{D}(i, 1) = \det \mathcal{E} = \det \hat{\mathcal{E}} \]

\[
= (-1)^{\text{ix}(r, \mathcal{M}^{(r)})+1} \det A \det \mathcal{C}^{(r)}(\text{ix}(r, \mathcal{M}^{(r)}), 1).
\]

Then we have

\[
n_r = (-1)^{i+1} \det \mathcal{D}(i, 1)/\det \mathcal{D}
\]

\[
= (-1)^{\text{ix}(r, \mathcal{M}^{(r)})+1} \frac{\det A \det \mathcal{C}^{(r)}(\text{ix}(r, \mathcal{M}^{(r)}), 1)}{\det A \det \mathcal{C}^{(r)}} = p_r^{(r)}.
\]

\( \square \)
We also have a similar lemma for the functions $\Phi_b$ of Definition 3.2.

**Lemma 3.12.** Let $\mathcal{N} = \{\tau, \ldots, k\}$. Denote the clutch matrices of a $\mathcal{N}$-clutch and a $\mathcal{M}(\tau)$-clutch by $\mathcal{D}$ and $\mathcal{C}(\tau)$ respectively. For $r \in \mathcal{N}$, define

$$F(r) = \det \mathcal{D}\{1, \ldots, r - \tau\}, \{1, \ldots, r - \tau\}$$

and for $r \in \mathcal{M}(\tau)$, define

$$E(r) = \det \mathcal{C}(\tau)\{1, \ldots, 1r, \mathcal{M}(\tau) - 1\}, \{1, \ldots, 1r, \mathcal{M}(\tau) - 1\}.$$

Note that $F(r)$ and $E(r)$ are both functions from $\mathbb{R}^{m+2}$ to $\mathbb{R}$. Also, for $w \in \mathcal{N}$ with $w \leq \max \mathcal{M}(\tau)$, let $\mu(w) = \min \{r \in \mathcal{M}(\tau) : r \geq w\}$. Then, for $w \in \mathcal{N}$ and for all $x = (x, y_0, \ldots, y_m) \in D(\delta)$ with $y_{r+1} = \cdots = y_k = 0$, there exists a function $C_w : \mathbb{R}^{m+2} \to \mathbb{R}$ such that

$$F(w)(x) = \begin{cases} C_w(x)E(\mu(w))(x) & \text{if } w \leq \max \mathcal{M}(\tau), \\ C_w(x) & \text{if } w > \max \mathcal{M}(\tau). \end{cases}$$

Moreover, if the Independent Types Property for a phase of type $\tau$ is satisfied, then

$$C_w = \begin{cases} (-1)^{k-w+1} \mathcal{M}(\tau) + \mathcal{M}(\tau) & \text{if } w \leq \max \mathcal{M}(\tau), \\ (-1)^{k-w+1} \mathcal{M}(\tau) & \text{if } w > \max \mathcal{M}(\tau). \end{cases}$$

**Proof.** The proof is similar to the proof of Lemma 3.11. Again fix $(x, y_0, \ldots, y_m) \in D(\delta)$ with $y_{r+1} = \cdots = y_k = 0$ and note that all functions are evaluated at $(x, y_0, \ldots, y_m)$. Define $\mathcal{D}$ to be the matrix obtained by permuting the rows and columns of $\mathcal{D}$ according to $\sigma$ (where $\sigma$ is defined in the proof of Lemma 3.11). Let $a = |\mathcal{N}|$ and let $b = |\mathcal{N} \setminus \mathcal{M}(\tau)|$. Recall from the proof of Lemma 3.11 that

$$\mathcal{D} = \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{O} & \mathbf{C} \end{pmatrix}$$

where $\mathbf{A}$ is a $b \times b$ matrix (as given by (3.7)), $\mathbf{B}$ is a $b \times (a-b)$ matrix, $\mathbf{O}$ is an $(a-b) \times b$ zero matrix, and $\mathbf{C} = \mathcal{C}(\tau)$. Now fix $w \in \mathcal{N}$. Then

$$F(w) = \det \mathcal{D}\{\sigma(1), \ldots, \sigma(w - \tau)\}, \{\sigma(1), \ldots, \sigma(w - \tau)\}.$$

Let $\xi_w = \{|\tau, \ldots, w - 1\} \setminus \mathcal{M}(\tau)$ and let $\chi_w = \{|\tau, \ldots, w - 1\} \cap \mathcal{M}(\tau)$. Then

$$\sigma(\{1, \ldots, w - \tau\}) = \{1, \ldots, \xi_w\} \cup \{b + 1, \ldots, b + \chi_w\}.$$

Therefore

$$F(w) = \det \begin{pmatrix} \hat{\mathbf{A}} & \hat{\mathbf{B}} \\ \hat{\mathbf{O}} & \hat{\mathbf{C}} \end{pmatrix}$$

where

- $\hat{\mathbf{A}} = \mathbf{A}(\{1, \ldots, \xi_w\}, \{1, \ldots, \xi_w\})$,
- $\hat{\mathbf{B}} = \mathbf{B}(\{1, \ldots, \xi_w\}, \{1, \ldots, \chi_w\})$. 

14
Thus, for a phase of type \( \tau \) expected change in the random variable \( y \) of the deprioritised algorithm. Considerations by considering the expected change in the random variables due to an operation \( \hat{y} \) under certain conditions, the scaled random variables \( \hat{y} \) are defined piecewise, with one piece per phase. Consider a phase of type \( \tau \). The expected change in the random variable \( Y_t \) due to the operation at step \( t \) (occurring in the main subphase of a phase of type \( \tau \)) is

\[
\mathbb{E}(Y_t(t+1) - Y_t(t) \mid \text{op}_{t+1}) = \sum_{r=0}^{k} \mathbb{E}(Y_t(t+1) - Y_t(t) \mid \tau(\text{op}_{t+1}) = r) \mathbb{P}(\tau(\text{op}_{t+1}) = r)
\]

\[
= \sum_{r=0}^{k} f_{i,r}(t)p_{r}^{(\tau)}(t) + o(1).
\]

Thus, for a phase of type \( \tau \), we use the differential equations

\[
\frac{dy_t}{dx} = \sum_{r=0}^{k} f_{i,r}(x)p_{r}^{(\tau)}(x)
\]

for \( 0 \leq i \leq m \). Solving these differential equations with appropriate initial conditions we obtain the functions \( \hat{y} \) during the corresponding phase.

The domain on which we solve the differential equations is defined as the intersection of three subsets of \( \mathbb{R}^{m+2} \). Let \( C \) be a constant such that \( Cn \) is an upper bound on the number of operations performed by the deprioritised algorithm. Recall the constant \( M \)
and polynomial $H$ from Properties 3.1. Earlier in (3.3) we defined the domain $D(\delta)$. On $D(\delta)$ the functions $f_{i,r}$ are Lipschitz, which can be seen by considering the partial derivatives of $f_{i,r}$. The parameter $\delta$ expresses how close to the end of the deprioritised algorithm we can approximate the random variables; while the condition $y_0 > \delta$ allows the preprocessing subphases to be performed.

Recall that $C^{(r)}$ is the clutch matrix for an $\mathcal{M}^{(r)}$-clutch. Now for $b = 1, \ldots, |\mathcal{M}^{(r)}|$, let

$$q_b^{(r)} = (-1)^{|\mathcal{M}^{(r)}|} b \det C^{(r)}(b, 1).$$

(3.9)

We then define

$$L^{(r)}(\delta) = \{(x, y_0, \ldots, y_m) : (-1)^{|\mathcal{M}^{(r)}|} \det C^{(r)} > \delta, q_b^{(r)} > 0 \ (b = 1, \ldots, |\mathcal{M}^{(r)}|)\}.$$ 

By Lemma 3.4, the definition of $L^{(r)}(\delta)$ ensures that, on $D(\delta) \cap L^{(r)}(\delta)$, the functions $p_i^{(r)}$ define a probability distribution. Moreover, applying a theorem of Hurewicz [8, Chapter 2] the differential equations are Lipschitz on $L^{(r)}(\delta)$ for all $\delta > 0$ (see [7, Section 4.3.1]). Indeed, all the functions we consider when analysing a deprioritised algorithm are Lipschitz on the domains $L^{(r)}(\delta)$.

Now for $b = 2, \ldots, |\mathcal{M}^{(r)}|$, we let

$$E_b^{(r)} = (-1)^{|\mathcal{M}^{(r)}|} b^{+1} \det C^{(r)}(\{1, \ldots, b - 1\}, \{1, \ldots, b - 1\}).$$

(3.10)

Then we define $A^{(r)} = \{(x, y_0, \ldots, y_m) : y_r > 0, E_b^{(r)} > 0 \ (b = 2, \ldots, |\mathcal{M}^{(r)}|)\}$. The domain $A^{(r)}$ defines the boundaries of a phase of type $\tau$. We solve the differential equations (3.8) on the domain $V^{(r)}(\delta) = D(\delta) \cap L^{(r)}(\delta) \cap A^{(r)}$.

**Definition 3.13 (Functions $\hat{y}_i$).** Let $x_0 = 0$ and define $\hat{y}_i(x_0) = \lim_{n \to \infty} Y_i(0)/n$ for $i = 0, \ldots, m$ (by (vii) Properties 3.1, these limits exist). Fix $\delta > 0$ and assume that $\tau_i$ is given and satisfies $\hat{y}_{\tau_i+1}(x_0) = \ldots = \hat{y}_k(x_0) = 0$. Denote the closure of $V^{(\tau)}(\delta)$ by $\overline{V^{(\tau)}(\delta)}$.

Assume that $j \geq 1$ and that $\tau_j$ is defined. If $(x_{j-1}, y_0(x_{j-1}), \ldots, y_m(x_{j-1})) \not\in \overline{V^{(\tau)}(\delta)}$, then there is no phase $j$, so set $K = j - 1$ and finish the definition of $\hat{y}_i$. Otherwise let $y_0^{(j)}, \ldots, y_m^{(j)}$ be the solutions to the differential equations (3.8) with $\tau = \tau_j$ and initial conditions $y_i^{(j)}(x_{j-1}) = \hat{y}_i(x_{j-1})$ (for $i = 0, \ldots, m$) on some open set containing $\overline{V^{(\tau)}(\delta)}$ for which the Lipschitz property still holds (it is easy to check such a set exists). Then define $x_j$ to be the infimum of those $x > x_{j-1}$ for which $(x, y_0^{(j)}(x), \ldots, y_m^{(j)}(x)) \not\in V^{(\tau)}(\delta)$. We extend $\hat{y}_i$ by defining $\hat{y}_i(x) = y_i^{(j)}(x)$ for $x \in [x_{j-1}, x_j]$. Whether there is a next phase, and if so its type, is determined by the following steps:

(i) if $(x_j, y_0^{(j)}(x_j), \ldots, y_m^{(j)}(x_j))$ lies on the boundary of $D(\delta)$, then there is no phase $j + 1$, so set $K = j$ and finish the definition of $\hat{y}_i$; otherwise

(ii) if $(x_j, y_0^{(j)}(x_j), \ldots, y_m^{(j)}(x_j))$ lies on the boundary of

$$\{(x, y_0, \ldots, y_m) : E_b^{(\tau_j)} > 0 \text{ for } b = 2, \ldots, |\mathcal{M}^{(\tau_j)}|\},$$

then we set $\tau_{j+1}$ to be the maximum $r \in \mathcal{M}^{(\tau_j)}$ such that $E_b^{(\tau_j)}(x_j) = 0$ where $b = \text{ix}(r, \mathcal{M}^{(\tau_j)})$; otherwise
(iii) if \( y^{(j)}_{\tau_j}(x_j) = 0 \), then we set \( \tau_{j+1} \) to be the maximum \( r \in \{0, \ldots, k\} \) such that \( y^{(j)}_{\tau_r}(x_j) > 0 \); otherwise

(iv) there is no phase \( j+1 \), so set \( K = j \) and finish the definition of \( \hat{y}_i \).

Note that for phases after the first phase, the phase type is determined by the functions \( y^{(j)}_i \) which correspond to the previous phase. Also, the above definition allows for a more general sequence of phases than allowed in Wormald’s previous analysis [13]. In particular, we allow multiple phases of the same type and we allow phases of type 0. Note that a phase of type 0 is different from a preprocessing subphase, since operations of types greater than 0 may (and usually do) occur. The next corollary gives some useful properties of the phase types defined by Definition 3.13.

**Corollary 3.14.** For some \( j \geq 2 \), assume that \( \tau_j \) is defined according to Definition 3.13. Then the following are true:

- the point \( (x_{j-1}, \hat{y}_0(x_{j-1}), \ldots, \hat{y}_m(x_{j-1})) \) lies in \( D(\delta) \),
- if \( \tau_j > \tau_{j-1} \), then \( \tau_j \in \mathcal{M}^{(\tau_j-1)} \) and for \( \beta = \text{ix}(\tau_j, \mathcal{M}^{(\tau_j-1)}) \) we have

\[
E^{(\tau_j-1)}_\beta(x_{j-1}) = 0 \quad \text{and} \quad E^{(\tau_j-1)}_b(x_{j-1}) > 0
\]

for \( b = \beta + 1, \ldots, |\mathcal{M}^{(\tau_j-1)}| \), and

- if \( \tau_j < \tau_{j-1} \), then \( E^{(\tau_j-1)}_b(x_{j-1}) > 0 \) for \( b = 2, \ldots, |\mathcal{M}^{(\tau_j-1)}| \).

We also note the following useful properties of the differential equations and the functions \( \hat{y}_i \).

**Lemma 3.15.** Let \( dy_i/dx \) be defined by (3.8). For all \((x, y_0, \ldots, y_m) \in D(\delta)\), we have

\[
\frac{dy_i}{dx}(x, y_0, \ldots, y_m) = 0
\]

for \( i \in \mathcal{M}^{(\tau)} \backslash \{\tau\} \). Furthermore, if \( y_{\tau+1} = \cdots = y_k = 0 \) also, then for \( i = \tau + 1, \ldots, k \) we have

\[
\frac{dy_i}{dx}(x, y_0, \ldots, y_m) = 0.
\]

**Proof.** Let \( \mathcal{M}^{(\tau)} = \{w_1, \ldots, w_a\} \) where \( w_1 < \cdots < w_a \). Recall that \( C^{(\tau)} \) is the clutch matrix for an \( \mathcal{M}^{(\tau)} \)-clutch. Then

\[
\det C^{(\tau)} \frac{dy_i}{dx} = \det A \quad \text{where} \quad A = \begin{pmatrix}
    f_{i,w_1} & f_{i,w_2,w_1} & \cdots & f_{i,w_a,w_1} \\
    \vdots & \vdots & \ddots & \vdots \\
    f_{i,w_a} & f_{i,w_2,w_a} & \cdots & f_{i,w_a,w_a}
\end{pmatrix}.
\]

For \( i \in \mathcal{M}^{(\tau)} \) with \( i > \tau \), the matrix \( A \) has two equal columns, and so \( dy_i/dx = 0 \). While for \( i \in \{\tau + 1, \ldots, k\} \backslash \mathcal{M}^{(\tau)} \), by Lemma 3.7, when \( y_{\tau+1} = \cdots = y_k = 0 \) the first column of \( A \) is all zero. So again \( dy_i/dx = 0 \). \( \square \)
From Lemma 3.15 and the definition of the phase types (from Definition 3.13), we see that for each phase $j$, the functions $\hat{y}_{\tau+1}, \ldots, \hat{y}_k$ are zero on the interval $[x_{j-1}, x_j]$. It is also useful to note that $\hat{y}_{\tau_j}(x_{j-1}) = 0$ when $\tau_j > \tau_{j-1}$.

**Lemma 3.16 ([7, Lemma 4.3.5]).** For $j = 1, \ldots, K$, where $K$ is defined in Definition 3.13, we have
\[ \hat{y}_{\tau_j+1}(x) = \cdots = \hat{y}_k(x) = 0 \]
for all $x \in [x_{j-1}, x_j]$. Therefore, for $j > 1$, if $\tau_j > \tau_{j-1}$, then $\hat{y}_{\tau_j}(x_{j-1}) = 0$.

### 3.2.1 The Deprioritised Algorithm

Finally, Algorithm 1 gives pseudo-code for the deprioritised algorithm based on a given prioritised algorithm satisfying Properties 3.1. As input, the deprioritised algorithm takes the type of phase one and a sequence of positive real numbers $\epsilon_1, \ldots, \epsilon_K$. These numbers are the (scaled) lengths of the preprocessing subphases. The number of phases and, except for phase one, their types are determined as in Definition 3.13. The domains $D(\delta)$ and $V^{(\tau)}(\delta)$ and the type distribution depend solely on the prioritised algorithm. Recall that the deprioritised algorithm uses the same operations and random variables as the prioritised algorithm.

**Algorithm 1** The deprioritised algorithm

Set $G := G_0$;

for $j = 1, \ldots, K$ do

  # The preprocessing subphase of phase $j$

  for $v = 1, \ldots, \lfloor \epsilon_j n \rfloor$ do

    if $(Y_{0}(G)/n, \ldots, Y_{m}(G)/n) \in D(0)$ then

      Perform an operation of type 0;

    else

      return fail;

    end if

  end for

  # The main subphase of phase $j$

  while $(Y_{0}(G)/n, \ldots, Y_{m}(G)/n) \in V^{(\tau)}(0)$ do

    Calculate the type distribution $p_{r(\tau)}(Y_{0}(G), \ldots, Y_{m}(G))$ for $r = 0, \ldots, k$;

    Select a type $r$ randomly using the type distribution;

    if $Y_r(G) = 0$ then

      return fail;

    else

      Perform an operation of type $r$;

    end if

  end while

end for
4 Analysing deprivatized algorithms

Each phase of a deprivatized algorithm is analysed with Theorem 4.1, stated below. This theorem is very similar to two theorems by Wormald [13, Theorem 3], [12, Theorem 5.1].

Note that for a some $W \subseteq \mathbb{R}^{a+1}$ and random variables $Z_1, \ldots, Z_a$, we define the stopping time $T_W$ to be the minimum $t$ such that $(t/n, Z_1(t)/n, \ldots, Z_a(t)/n) \notin W$.

**Theorem 4.1.** Let $a$ be a fixed positive integer. For $1 \leq \ell \leq a$, let $Z_\ell$ be a random variable defined on a discrete time Markov process $\{G_t\}_{t \geq 0}$. Assume that $W \subset \mathbb{R}^{a+1}$ is open and bounded such that

(i) for some closed subset $U$ of $W$, a.a.s. $(0, Z_1(0)/n, \ldots, Z_a(0)/n) \in U$,

(ii) (Boundedness Hypothesis) for some constant $\beta$ we have

$$\max_{1 \leq \ell \leq a} |Z_\ell(t+1) - Z_\ell(t)| \leq \beta$$

for $t \geq 0$, and

(iii) (Trend Hypothesis) for some functions $F_\ell : \mathbb{R}^{a+1} \to \mathbb{R}$, which are Lipschitz on $W$ for all $1 \leq \ell \leq a$, and for some $\lambda = \lambda(n) = o(1)$, we have

$$|\mathbb{E}(Z_\ell(t+1) - Z_\ell(t) \mid G_0, \ldots, G_t) - F_\ell(t/n, Z_1(t)/n, \ldots, Z_a(t)/n)| \leq \lambda$$

for $t < T_W$ and $1 \leq \ell \leq a$.

Then the following are true:

(a) For $(0, \hat{z}_1, \ldots, \hat{z}_a) \in W$ the system of differential equations

$$\frac{dz_\ell}{dx} = F_\ell(x, z_1, \ldots, z_a) \quad (4.1)$$

has a unique solution in $W$ for $z_\ell : \mathbb{R} \to \mathbb{R}$ such that $z_\ell(0) = \hat{z}_\ell$ for $1 \leq \ell \leq a$ and which extends to points arbitrarily close to the boundary of $W$.

(b) Asymptotically almost surely, for $\ell = 1, \ldots, a$, we have $Z_\ell(t) = nz_\ell(t/n) + o(n)$ uniformly for $0 \leq t \leq \sigma n$, where $(x, z_1(x), \ldots, z_a(x))$ is the solution to (4.1) with initial conditions $(0, Z_1(0)/n, \ldots, Z_a(0)/n)$ and $\sigma = \sigma(n)$ is the supremum of those $x$ to which the solution can be extended before being within $\ell^\infty$-distance $\mu$ of the boundary of $W$, for some $\mu > \lambda$ with $\mu = o(1)$.

We now describe how Theorem 4.1 is applied to a preprocessing subphase. We also use Theorem 4.1 to analyse main subphases. Consider phase $j$ of the deprivatized algorithm. Set the scaled length of the preprocessing subphase to be $\epsilon_j$ for some $\epsilon_j > 0$. Then $t_{j-1} = \lfloor nx_{j-1} \rfloor$ is the end of phase $j-1$ and the start of phase $j$, and $t'_j = t_{j-1} + \lfloor n\epsilon_j \rfloor$ is the end of the preprocessing subphase of phase $j$ and the start of the main subphase. Later we take $\epsilon_j = \epsilon_j(n) = o(1)$ and show that the effect of the preprocessing subphases on the behaviour of the random variables is negligible.

For the preprocessing subphase, we apply Theorem 4.1 to the random variables $Z_i(t) = Y_i(t_j-1 + t)$ with the functions $F_i = f_{i,0}$ on the domain

$$W_{\delta, \epsilon_j} = D(\delta) \cap \{(x, y_0, \ldots, y_m) : -\delta < x < 2\epsilon_j\}.$$
Recall that during a preprocessing subphase, only type 0 operations are performed. The Boundedness and Trend Hypotheses are satisfied by Properties 3.1. Hypothesis (i) is satisfied whenever we have

\[(x_{j-1}, \hat{y}_0(x_{j-1}), \ldots, \hat{y}_m(x_{j-1})) \in D(\delta) \tag{4.2}\]

and

\[|(x_{j-1}, \hat{y}_0(x_{j-1}), \ldots, \hat{y}_m(x_{j-1})) - (x_{j-1}, Y_0(t_{j-1})/n, \ldots, Y_m(t_{j-1})/n)| = o(1). \tag{4.3}\]

We will only analyse phase \(j\) when the above two conditions hold. Note that, for \(j > 1\), equation (4.2) holds whenever \(\tau_j\) is defined according to Definition 3.13 (see Corollary 3.14).

Now define \((x, z_0^{(p)}(x), \ldots, z_m^{(p)}(x))\) to be the solution to the system of differential equations \(dy_i/dx = f_{i,0}\) with initial conditions \((0, Y_0(t_{j-1})/n, \ldots, Y_m(t_{j-1})/n)\) on \(W_{\delta,\epsilon_j}\).

**Lemma 4.2.** If (4.2) and (4.3) hold, then for all \(\epsilon_j\) sufficiently small, a.a.s. we have

(i) \(Y_i(t'_{j-1}) = n z_i^{(p)}(\epsilon_j) + o(n)\) for \(i = 0, \ldots, m\), and

(ii) for \(x \in [0, \epsilon_j]\), the point \((x, z_0^{(p)}(x), \ldots, z_m^{(p)}(x))\) is bounded away from the boundary of \(W_{\delta,\epsilon_j}\).

**Proof.** By applying Theorem 4.1 as described above, we conclude that for \(i = 0, \ldots, m\), a.a.s. we have

\[Y_i(t_{j-1} + t) = n z_i^{(p)}(t/n) + o(n) \tag{4.4}\]

for \(0 \leq t \leq \sigma n\), where \(\sigma\) is the supremum of those \(x\) to which \((x, z_0^{(p)}(x), \ldots, z_m^{(p)}(x))\) can be extended before being within some distance \(d(n) = o(1)\) of the boundary of \(W_{\delta,\epsilon_j}\).

We now show that \(\sigma \to 2\epsilon_j\) for sufficiently small \(\epsilon_j\). Consider the condition \(H > \delta\) from the definition of \(D(\delta)\) (and hence the definition of \(W_{\delta,\epsilon_j}\)). Recall that \(H\) is a polynomial and so is Lipschitz on a bounded domain. By (4.2), (4.3), and the Lipschitz property of \(H\) we have \(H((t_{j-1} + t)/n, Y_0(t_{j-1})/n, \ldots, Y_m(t_{j-1})/n) > \delta\) for sufficiently large \(n\). Since each operation changes the random variables by only a constant, and as \(H\) is Lipschitz, we have

\[H((t_{j-1} + t)/n, Y_0(t_{j-1} + t)/n, \ldots, Y_m(t_{j-1} + t)/n) = H(t_{j-1}/n, Y_0(t_{j-1})/n, \ldots, Y_m(t_{j-1})/n) + O(t/n)\]

for \(t = 0, \ldots, [2n\epsilon_j]\). Therefore, for all \(\epsilon_j\) sufficiently small and for \(n\) sufficiently large, we have \(H((t_{j-1} + t)/n, Y_0(t_{j-1} + t)/n, \ldots, Y_m(t_{j-1} + t)/n)\) bounded above \(\delta\) for \(t = 0, \ldots, [2n\epsilon_j]\). Thus, from (4.4), a.a.s. \(H(x, z_0^{(p)}(x), \ldots, z_m^{(p)}(x))\) is bounded above \(\delta\) for \(x \in [0, \epsilon_j]\). Similar reasoning applies for the other conditions defining \(W_{\delta,\epsilon_j}\) except \(x < 2\epsilon_j\). Therefore \(\sigma \to 2\epsilon_j\) and the result follows.

To analyse the preprocessing subphase of phase \(j\), we also consider the solution \((x, \hat{y}_0^{(p)}(x), \ldots, \hat{y}_m^{(p)}(x))\) to the system of differential equations \(dy_i/dx = f_{i,0}\) with initial conditions \((x_{j-1}, \hat{y}_0(x_{j-1}), \ldots, \hat{y}_m(x_{j-1}))\). By a standard result in the theory of differential equations [13, Lemma 1], when (4.3) holds we have

\[|(z_0^{(p)}(x), \ldots, z_m^{(p)}(x)) - (\hat{y}_0^{(p)}(x_{j-1} + x), \ldots, \hat{y}_m^{(p)}(x_{j-1} + x))| = o(1),\]
provided the functions are defined. Thus, considering part (ii) of Lemma 4.2, we have the following result.

**Lemma 4.3 ([7, Lemma 5.2.3]).** If (4.2) and (4.3) hold, then for all $\epsilon_j$ sufficiently small, a.a.s. we have

\[
\left| (z_0^{(p)}(\epsilon_j), \ldots, z_m^{(p)}(\epsilon_j)) - (y_0^{(p)}(x'_{j-1}), \ldots, y_m^{(p)}(x'_{j-1})) \right| = o(1).
\]

We now determine hypotheses that allow a phase of a deprioritised algorithm to be analysed. Consider phase $j$ of type $\tau_j$. Each hypothesis for phase $j$ concerns one or more of the functions defining the domain $V^{(\tau_j)}(\delta)$. These are the functions in the set

\[
B^{(\tau_j)}(\delta) = \{ y_0 - \delta, x + \delta, C - x, H - \delta, y_{\tau_j}, (-1)^{|M^{(\tau_j)}|+1} \det C^{(\tau_j)} - \delta, 2M - y_i \ (i = 0, \ldots, m), y_i + 2M \ (i = 0, \ldots, m), q_b^{(\tau_j)} \ (b = 1, \ldots, |M^{(\tau_j)}|), E_b^{(\tau_j)} \ (b = 2, \ldots, |M^{(\tau_j)}|) \}
\]

where $q_b^{(\tau_j)}$ and $E_b^{(\tau_j)}$ are defined by (3.9) and (3.10) respectively. Notice that

\[
h(x, y_0, \ldots, y_m) > 0 \text{ for all } h \in B^{(\tau_j)}(\delta) \implies (x, y_0, \ldots, y_m) \in V^{(\tau_j)}(\delta).
\]

Moreover, a point $(x, y_0, \ldots, y_m)$ satisfying $h(x, y_0, \ldots, y_m) \geq 0$ for all $h \in B^{(\tau_j)}(\delta)$ lies in the closure of $V^{(\tau_j)}(\delta)$. From the definition of $\tilde{g}_i$ (Definition 3.13), we only analyse the deprioritised algorithm while the functions

\[
y_0 - \delta, x + \delta, C - x, H - \delta, 2M - y_i \ (i = 0, \ldots, m), y_i + 2M \ (i = 0, \ldots, m)
\]

are strictly positive. The remaining functions of $B^{(\tau_j)}(\delta)$ may be zero at some point of the analysis; so we also define

\[
\tilde{B}^{(\tau_j)}(\delta) = \{ (-1)^{|M^{(\tau_j)}|+1} \det C^{(\tau_j)} - \delta, y_{\tau_j}, q_b^{(\tau_j)} \ (b = 1, \ldots, |M^{(\tau_j)}|), E_b^{(\tau_j)} \ (b = 2, \ldots, |M^{(\tau_j)}|) \}
\]

To analyse a deprioritised algorithm, we require that phases have non-zero length. That is, we require that for some $c_1 > x_{j-1}$ we have $(x, y_0^{(j)}(x), \ldots, y_m^{(j)}(x)) \in V^{(\tau_j)}(\delta)$ for $x \in [x_{j-1}, c_1]$, where $y_i^{(j)}$ are the solutions to the differential equations for phase $j$ (see Definition 3.13). By (4.6), it is sufficient to show that, for all $h \in \tilde{B}^{(\tau_j)}(\delta)$, we have

\[
h(x, y_0^{(j)}(x), \ldots, y_m^{(j)}(x)) > 0 \text{ for } x \in [x_{j-1}, c_1].
\]

We consider a general setting. Assume that $y_0(x), \ldots, y_m(x)$ satisfy the differential equations $dy_i/dx = g_i (i = 0, \ldots, m)$ for some functions $g_i$ of $x, y_0, \ldots, y_m$. For a function $h$ of $x, y_0, \ldots, y_m$, the derivative of $h$ with respect to $x$ is given by $\sum_{i=0}^m \frac{\partial h}{\partial y_i} \cdot g_i$. There are two important cases: when $dy_i/dx = f_{i,0}$, which corresponds to a preprocessing subphase, and when $dy_i/dx$ is given by (3.8), which corresponds to a main subphase as well as the functions $\tilde{g}_i$. For the first case we denote the derivative using the differential operator $\Delta^{(p)}$ defined by

\[
\Delta^{(p)} h = \sum_{i=0}^m \frac{\partial h}{\partial y_i} \cdot f_{i,0}.
\]
In the second case we denote the derivative using the differential operator $\Delta^{(r)}$ defined by

$$\Delta^{(r)} h = \sum_{i=0}^{m} \left[ \frac{\partial h}{\partial y_i} \sum_{r=0}^{k} p_r^{(r)} f_i,r \right],$$

where $p_r^{(r)}$ are defined in Definition 3.3. For arbitrary differential equations we denote the corresponding differential operator by $\Delta_g$.

We are interested in showing that some of the functions in $B^{(r)}(\delta)$ are increasing, so we make the following definition.

**Definition 4.4 (Positive Growth).** A function $h$ of $(x, y_0, \ldots, y_m)$ has positive growth (of order $s$) at $(x^*, y_0^*, \ldots, y_m^*)$ (with respect to the differential equations $dy_i/dx = g_i$) if, for some non-negative integer $s$, we have

1. for $r = 0, \ldots, s$, the function $(\Delta_g)^rh$ exists and is continuous in $(x, y_0, \ldots, y_m)$ on an open neighbourhood containing $(x^*, y_0^*, \ldots, y_m^*)$,
2. $(\Delta_g)^a h(x^*, y_0^*, \ldots, y_m^*) = 0$ for $\alpha = 0, \ldots, s - 1$, and
3. $(\Delta_g)^s h(x^*, y_0^*, \ldots, y_m^*) > 0.$

We also say that a function has negative growth if the inequality in (iii) is reversed. For the particular cases we are interested in, that is, when $\Delta_g = \Delta^{(r)}$ or $\Delta_g = \Delta^{(s)}$, we prefer to say ‘during a preprocessing subphase’ and ‘during a phase of type $\tau$’ respectively, instead of ‘with respect to the differential equations $dy_i/dx = g_i$’. We will only use Definition 4.4 with $(x^*, y_0^*, \ldots, y_m^*) = (x_j, \dot{y}_0(x_j), \ldots, \dot{y}_m(x_j))$ for some $j$. So, instead of ‘at the point $(x_j, \dot{y}_0(x_j), \ldots, \dot{y}_m(x_j))$', we say ‘at the point $x_j$’. Finally we note that the functions of $B^{(r)}(\delta)$ are all rational functions in $x, y_0, \ldots, y_m$. As we consider these functions on domains that exclude their poles, condition (i) is always satisfied.

A function with positive or negative growth is bounded below on some interval.

**Lemma 4.5 (\cite[Lemma 5.3.2]{7}).** Let $h(x, y_0, \ldots, y_m)$ have positive or negative growth of order $s$ at $(x^*, y_0^*, \ldots, y_m^*)$ with respect to the differential equations $dy_i/dx = g_i$. If each $g_i$ is Lipschitz on some domain open domain $U$ containing $(x^*, y_0^*, \ldots, y_m^*)$ then define

$$h(x) = h(x, y_0(x), \ldots, y_m(x))$$

where $y_0(x), \ldots, y_m(x)$ are the solutions to the differential equations $dy_i/dx = g_i$ with initial conditions $(x^*, y_0^*, \ldots, y_m^*)$. For any strict lower bound $L$ on $(\Delta_g)^s h(x^*)$ there exists a constant $C_L > 0$ such that

$$h(x) > \frac{L(x - x^*)^s}{s!}$$

for $x \in (x^*, x^* + C_L]$.

Note that a function with positive growth is positive on some interval. So if each $h \in B^{(r)}(\delta)$ has positive growth at $x_{j-1}$ during a phase of type $\tau_j$, then, by Lemma 4.5 and (4.6), phase $j$ has non-zero length.
We also apply Theorem 4.1 to the main subphase of phase $j$. So we also define $(x,z_0^{(j)}(x),\ldots,z_m^{(j)}(x))$ to be the solution to the system of differential equations (3.8) (with $\tau = \tau_j$) for the initial conditions $(0,Y_0(t'_{j-1})/n,\ldots,Y_m(t'_{j-1})/n)$ on a subset of the domain $V^{(\tau_j)}(\delta)$ (which we give later). To prove Theorem 4.8 we also need to show that, for some $C > 0$, we have

$$(x,z_0^{(j)}(x),\ldots,z_m^{(j)}(x)) \in V^{(\tau_j)}(\delta) \text{ for } x \in [0,C].$$

By analysing the preprocessing subphase, we are able to show that

$$(0,z_0^{(j)}(0),\ldots,z_m^{(j)}(0)) = (0,Y_0(t'_{j-1})/n,\ldots,Y_m(t'_{j-1})/n) \in V^{(\tau_j)}(\delta).$$

We now determine conditions that, together with (4.8), imply (4.7).

Since we do not know $z_i^{(j)}(x'_{j-1})$ exactly, we cannot apply Lemma 4.5 to $z_i^{(j)}$ as we can to $y_i^{(j)}$. Instead we make the following definition. For any $h \in B^{(\tau_j)}(\delta)$ satisfying

(i) for some non-negative integer $s$, the function $h$ has positive growth of order $s$ at $x_{j-1}$ during a phase of type $\tau_j$, and

(ii) for $\alpha = 0,\ldots,s-1$, for some non-negative integer $s_\alpha$, the function $(\Delta^{(\tau_j)})^\alpha h$ has positive or negative growth of order $s_\alpha$ at $x_{j-1}$ during a preprocessing subphase, we let $B_\alpha$ (for $\alpha = 0,\ldots,s-1$) be a strict lower bound for

$$(\Delta^{(\tau_j)})^s(\Delta^{(\tau_j)})^\alpha h(x_{j-1},\hat{y}_0(x_{j-1}),\ldots,\hat{y}_0(x_{j-1})))$$

and let $L > 0$ be a strict lower bound for $(\Delta^{(\tau_j)})^s h(x_{j-1},\hat{y}_0(x_{j-1}),\ldots,\hat{y}_0(x_{j-1})))$, and define

$$\mathcal{L}_h(x,\epsilon_j) = \frac{Lx^s}{s!} + \sum_{\alpha=0}^{s-1} \frac{B_\alpha \epsilon_j^{s_\alpha} x^\alpha}{s_\alpha! \alpha!}.$$

Under certain conditions $\mathcal{L}_h(x,\epsilon_j)$ is a lower bound for $h(x,z_0^{(j)}(x),\ldots,z_m^{(j)}(x))$.

Lemma 4.6. If equations (4.2) and (4.3) hold and the function $\mathcal{L}_h(x,\epsilon_j)$ is defined for $h \in B^{(\tau_j)}(\delta)$, then for some constant $C_h > 0$ and for all $\epsilon_j$ sufficiently small, a.a.s. we have

$$h(x,z_0^{(j)}(x),\ldots,z_m^{(j)}(x)) > \mathcal{L}_h(x-x'_{j-1},\epsilon_j)$$

for $x \in (x_{j-1},x'_{j-1} + C_h]$.

Proof. For $\alpha = 0,\ldots,s-1$, let $B'_\alpha$ be a strict lower bound on

$$(\Delta^{(\tau_j)})^s(\Delta^{(\tau_j)})^\alpha h(x_{j-1},\hat{y}_0(x_{j-1}),\ldots,\hat{y}_0(x_{j-1}))$$

with $B'_\alpha > B_\alpha$. Then, since $(\Delta^{(\tau_j)})^\alpha h$ has positive or negative growth of order $s_\alpha$ at $x_{j-1}$ during a preprocessing subphase, by Lemma 4.5, we have

$$\mathcal{(\Delta^{(\tau_j)})^\alpha h(x,y_0^{(p)}(x),\ldots,y_m^{(p)}(x)) > B'_\alpha \frac{(x-x_{j-1})^{s_\alpha}}{s_\alpha!}}$$

for $x \in (x_{j-1},x_{j-1} + C_h]$ for some constant $C_h > 0$. 

23
Then using Lemma 4.3, Lemma 4.2, and the Lipschitz property of \((\Delta^{(\tau_j)})^\alpha h\), a.a.s. we have
\[
(\Delta^{(\tau_j)})^\alpha h(0, z_0^{(j)}(0), \ldots, z_m^{(j)}(0)) > B'_\alpha(\epsilon_j)^s_\alpha + o(1).
\]
By replacing \(B'_\alpha\) with \(B_\alpha\), we may drop the \(o(1)\) term. The result follows from the Fundamental Theorem of Calculus.

Thus by showing that \(L_h(x, \epsilon_j)\) is positive for all sufficiently small and positive \(x\) and \(\epsilon_j\), we may conclude that for some \(C > 0\) we have \(h(x, z_0^{(j)}(x), \ldots, z_m^{(j)}(x)) > 0\) for \(x \in [0, C]\). The most important cases for \(L_h(x, \epsilon_j)\) are given in the next lemma.

**Lemma 4.7.** For a function \(h\), define \(s_0, \ldots, s_{s-1}\) and \(B_\alpha\) for \(\alpha = 0, \ldots, s_{s-1}\), as above and assume that \(L_h(x, \epsilon_j)\) is defined. If (i) \(s = 0\), or (ii) \(s = 1\) and \(B_0 > 0\), or (iii) \(s = 2, s_0 = 1, s_1 = 1\), and \(B_0 > 0\), or (iv) \(B_\alpha > 0\) for \(\alpha = 0, \ldots, s-1\), then \(L_h(x, \epsilon_j)\) is positive for all sufficiently small and positive \(x\) and \(\epsilon\).

We can now give the hypotheses that allow phase \(j\) of the deprioritised algorithm to be analysed.

**Hypotheses: to be satisfied for each phase**

Let \(j\) be the phase number.

(A) There is a phase \(j\) according to Definition 3.13. That is, \(\tau_j\) is defined and
\[
(x_{j-1}, \hat{y}_0(x_{j-1}), \ldots, \hat{y}_m(x_{j-1})) \in \overline{V^{(\tau_j)}(\delta)},
\]
where \(\overline{V^{(\tau_j)}(\delta)}\) is the closure of \(V^{(\tau_j)}(\delta)\).

(B) Each function in the set \(\hat{B}^{(\tau_j)}(\delta)\) has positive growth at \(x_{j-1}\) during a phase of type \(\tau_j\).

(C) For \(i \in M^{(\tau_j)}\), the function \(y_i\) has positive growth at \(x_{j-1}\) during a preprocessing subphase.

(D) Each
\[
q \in \{(-1)^{|M^{(\tau_j)}|+1} \det C^{(\tau_j)} - \delta, q_b^{(\tau_j)} \ (b = 1, \ldots, |M^{(\tau_j)}|)\}
\]
has positive growth at \(x_{j-1}\) during a preprocessing subphase.

(E) For \(b = 2, \ldots, |M^{(\tau_j)}|\), the function \(E_b^{(\tau_j)}\) has positive growth at \(x_{j-1}\) during a preprocessing subphase.

(F) For each \(h \in \hat{B}^{(\tau_j)}(\delta)\), the function \(L_h(x, \epsilon_j)\) is defined and positive for all sufficiently small and positive \(x\) and \(\epsilon_j\).
Hypothesis (A) allows the functions $\hat{y}_i$ to be defined on $[x_{j-1}, x_j]$. We show that phase $j$ has non-zero length using (B). By (C) and (D), we can apply Theorem 4.1 to the main subphase of phase $j$: using (C) we show that an operation of the selected type can be performed and using (D) we show that the Trend Hypothesis of Theorem 4.1 is satisfied. Hypothesis (E), together with (C), ensures that the phase does not end immediately. The last hypothesis, (F), allows us to show that the functions $z^{(j)}_i$, obtained by applying Theorem 4.1 to the main subphase of phase $j$, do no stop approximating the scaled random variables immediately; thus we are able to show that the functions $z^{(j)}_i$ approximate the functions $\hat{y}_i$.

4.1 The Theorem

We now state the major theorem of this paper. Note that the conclusion of this theorem only concerns the random process defined by the deprioritised algorithm.

Theorem 4.8. Let $\mathcal{P}$ be an algorithm satisfying Properties 3.1 and recall $\hat{y}_0, \ldots, \hat{y}_m$, $K$, and $x_K$ from Definition 3.13. Recall that $\hat{y}_i(x_0) = \lim_{n \to \infty} Y_i(0)/n$. If

$$(x_0, \hat{y}_0(x_0), \ldots, \hat{y}_m(x_0)) \in D(\delta) \quad (4.9)$$

and hypotheses (A)-(F) are satisfied for $j = 1, \ldots, K$, then, for the random process $\{G_t\}$ generated by the deprioritised algorithm based on $\mathcal{P}$, a.a.s. we have

$$Y_i(G_t) = n\hat{y}_i(t/n) + o(n) \quad \text{for } i = 0, \ldots, m,$$

uniformly for $t = 0, \ldots, \lfloor nx_K \rfloor$.

We prove Theorem 4.8 by induction on the number of phases. The case $K = 0$ holds by the definition of $(x_0, \hat{y}_0(x_0), \ldots, \hat{y}_m(x_0))$ and (vii) of Properties 3.1. Now assume that, for some $j \leq K$, the conclusion of Theorem 4.8 holds for $t = 0, \ldots, \lfloor nx_{j-1} \rfloor$. The proof of the inductive step is done in three parts.

**Part I: Showing $x_j > x_{j-1}$.**

Let $\overline{V}^{(\tau_j)}(\delta)$ be the closure of $V^{(\tau_j)}(\delta)$. By hypothesis (A) and Definition 3.13, we have

$$(x_{j-1}, \hat{y}_0(x_{j-1}), \ldots, \hat{y}_m(x_{j-1})) \in \overline{V}^{(\tau_j)}(\delta).$$

The functions in the set $\mathcal{B}^{(\tau_j)}(\delta) \backslash \overline{\mathcal{B}}^{(\tau_j)}(\delta)$ have positive growth of order 0 at $x_{j-1}$ by hypothesis (A) and Corollary 3.14 if $j > 1$, or by (4.9) if $j = 1$. So, together with (B), we may apply Lemma 4.5 to each $h \in \mathcal{B}^{(\tau_j)}(\delta)$. So for each $h \in \mathcal{B}^{(\tau_j)}(\delta)$, there exists a $c_h > 0$ such that $h(x, y^{(j)}_0(x), \ldots, y^{(j)}_m(x)) > 0$ for $x \in (x_{j-1}, x_{j-1} + c_h]$. Thus, by (4.6), we have

$$x_j > x_{j-1} + \min_{h \in \mathcal{B}^{(\tau_j)}(\delta)} c_h > x_{j-1}.$$
PART II: The Preprocessing Subphase.

We will apply Theorem 4.1 to the main subphase of phase \( j \) with the random variables \( Z_i(t) = Y_i(t'_{j-1} + t) \) on the domain

\[
U^{(\tau_j)}(\delta) = V^{(\tau_j)}(\delta) \cap \{ (x, y_0, \ldots, y_m) : y_i > 0 \text{ for } i \in M^{(\tau_j)} \setminus \{ \tau_j \} \}.
\]

Recall that \( x'_{j-1} = x_{j-1} + \epsilon_j \) and that \( t'_{j-1} = \lceil nx'_{j-1} \rceil \) is the end of the preprocessing subphase. To satisfy hypothesis (i) of Theorem 4.1, we need to show that a.a.s. the point \((t'_{j-1}/n, Y_0(t'_{j-1})/n, \ldots, Y_m(t'_{j-1})/n)\) lies in \(U^{(\tau_j)}(\delta)\) and is a distance of at least a constant from the boundary of \(U^{(\tau_j)}(\delta)\).

Equation (4.2) holds by Corollary 3.14 and (A) (for \( j > 1 \)), or by (4.9) (for \( j = 1 \)). While equation (4.3) holds by the inductive hypothesis for \( j > 1 \), and by (vii) of Properties 3.1 for \( j = 1 \). So applying Lemma 4.3, for all sufficiently small \( \epsilon_j \), a.a.s. we have

\[
y_i^{(p)}(x'_{j-1}) = z_i^{(p)}(\epsilon_j) + o(1) \tag{4.10}
\]

for \( i = 0, \ldots, m \). Thus, by Lemma 4.2 (ii), for all sufficiently small \( \epsilon_j \), a.a.s. the point \((x'_{j-1}, y_0^{(p)}(x'_{j-1}), \ldots, y_m^{(p)}(x'_{j-1}))\) lies in \(D(\delta)\) and is a distance of at least a constant from the boundary of \(D(\delta)\).

As hypotheses (C), (D), and (E) hold, we may apply Lemma 4.5. Thus, by (4.6), for all \( \epsilon_j \) sufficiently small, a.a.s. the point \((x'_{j-1}, y_0^{(p)}(x'_{j-1}), \ldots, y_m^{(p)}(x'_{j-1}))\) lies in \(U^{(\tau_j)}(\delta)\) and is a distance of at least a constant from the boundary of \(U^{(\tau_j)}(\delta)\). From (4.10) and Lemma 4.2 (i), for all \( \epsilon_j \) sufficiently small, a.a.s. we have \(y_i^{(p)}(x'_{j-1}) = Y_i(t'_{j-1})/n + o(1)\) for \( i = 0, \ldots, m \).

Thus, for all \( \epsilon_j \) sufficiently small, a.a.s. the point \((t'_{j-1}/n, Y_0(t'_{j-1})/n, \ldots, Y_m(t'_{j-1})/n)\) lies in \(U^{(\tau_j)}(\delta)\) and is a distance of at least a constant from the boundary of \(U^{(\tau_j)}(\delta)\). Thus hypothesis (i) of Theorem 4.1 is satisfied.

PART III: The Main Subphase

We now apply Theorem 4.1 to the random variables \( Z_i(t) = Y_i(t'_{j-1} + t) \) with functions \( E_i \) given by the right hand side of (3.8) (with \( \tau = \tau_j \)) on the domain \(U^{(\tau_j)}(\delta)\).

Hypothesis (i) is satisfied by Part II and the Boundedness hypothesis is satisfied by part (v) of Properties 3.1. The definition of \(U^{(\tau_j)}(\delta)\) ensures that operations of types from \(M^{(\tau_j)}\) can be performed and that, by Lemma 3.4, the functions \(p_0^{(\tau_j)}, \ldots, p_k^{(\tau_j)}\) define a probability distribution. Thus the Trend hypothesis follows from part (iii) of Properties 3.1, the definition of \(U^{(\tau_j)}(\delta)\), the definition of the deprioritised algorithm, and as the Lipschitz conditions are satisfied.

Recall that \((x, z_0^{(j)}(x), \ldots, z_m^{(j)}(x))\) is the solution to the system of differential equations (3.8) (with \( \tau = \tau_j \)) for the initial conditions \(z_i^{(j)}(0) = Y_i(t'_{j-1})/n\), on the domain \(U^{(\tau_j)}(\delta)\). From Theorem 4.1, a.a.s. we have

\[
Y_i(t'_{j-1} + t) = nz_i^{(j)}(t/n) + o(n) \tag{4.11}
\]

uniformly for \( 0 \leq t \leq \sigma n \), where \( \sigma \) is the supremum of those \( x \) for which the solution \((x, z_0^{(j)}(x), \ldots, z_m^{(j)}(x))\) can be extended before being within some distance \(d(n) = o(1)\) of the boundary of \(U^{(\tau_j)}(\delta)\). Note that \( \sigma \) depends on \( \epsilon_j \).
First, by Lemma 3.15, for \( i \in \mathcal{M}^{(\tau_j)} \setminus \{\tau_j\} \) we have \( z_i^{(j)}(x) = z_i^{(j)}(0) = Y_i(t_{j-1}' - 1)/n > 0 \) for all \( x \) for which the solutions are defined. Therefore

\[
(x, z_0^{(j)}(x), \ldots, z_m^{(j)}(x)) \in V^{(\tau_j)}(\delta) \implies (x, z_0^{(j)}(x), \ldots, z_m^{(j)}(x)) \in U^{(\tau_j)}(\delta).
\] (4.12)

Now \(|(z_0^{(j)}(0), \ldots, z_m^{(j)}(0)) - (\hat{y}_0(x_{j-1}), \ldots, \hat{y}_m(x_{j-1}))| = O(\epsilon_j)\) by part (v) of Properties 3.1. Taking \( \epsilon_j = \epsilon_j(n) = o(1) \) and applying a standard result [13, Lemma 1] as earlier we have

\[
|(z_0^{(j)}(x), \ldots, z_m^{(j)}(x)) - (\hat{y}_0(x_{j-1} + x), \ldots, \hat{y}_m(x_{j-1} + x))| = o(1)
\] (4.13)

uniformly for \( x \in [0, \min\{x_1^*, x_2^*\}] \), where \( x_1^* \) is the infimum of those \( x > 0 \) for which \( (x, z_0^{(j)}(x), \ldots, z_m^{(j)}(x)) \notin V^{(\tau_j)}(\delta) \) and \( x_2^* \) is the infimum of those \( x > 0 \) for which \( (x, \hat{y}_0(x_{j-1} + x), \ldots, \hat{y}_m(x_{j-1} + x)) \notin V^{(\tau_j)}(\delta) \). Note that by definition we have \( \sigma < x_1^* \) and \( x_1^* = x - x_{j-1} \).

Next we show that \( x_j - x_{j-1} \) can be at most \( o(1) \) larger than \( \sigma \). By (F) and Lemma 4.6, for some constant \( C > 0 \) and for all \( h \in B^{(\tau_j)}(\delta) \) we have \( h(x, z_0^{(j)}(x), \ldots, z_m^{(j)}(x)) > 0 \) for \( x \in [0, C] \). Therefore a.a.s. \( \sigma > C \); note that \( C \) does not depend on \( \epsilon_j \). Let \( D = \min\{C, \min_{k \in B^{(\tau_j)}(\delta)} c_k\} \) where \( c_k \) is defined in Part I.

Now condition on the event \( \sigma > D \). Consider \( \kappa > 0 \) with \( x_j - x_{j-1} - \kappa > D \) (such \( \kappa \) exist by Part I). Assume that \( x_j - x_{j-1} - \kappa > \sigma \) for infinitely many \( n \). From (4.13) we have

\[
|(z_0^{(j)}(\sigma), \ldots, z_m^{(j)}(\sigma)) - (\hat{y}_0(\sigma), \ldots, \hat{y}_m(\sigma))| = o(1)
\]

since \( \sigma < x_1^* \) and \( \sigma < x_j - x_{j-1} = x_2^* \). By definition, as \( x \to \sigma \), the solution \((x, z_0^{(j)}(x), \ldots, z_m^{(j)}(x))\) approaches the boundary of \( V^{(\tau_j)}(\delta) \). Hence the distance from \((x, \hat{y}_0(x_{j-1} + x), \ldots, \hat{y}_m(x_{j-1} + x))\) to the boundary of \( V^{(\tau_j)}(\delta) \) is bounded above by a function that tends to zero as \( n \) tends to infinity. However \( \sigma \in [D, x_j - x_{j-1} - \kappa] \) and \((x, \hat{y}_0(x_{j-1} + x), \ldots, \hat{y}_m(x_{j-1} + x))\) is bounded away from the boundary of \( V^{(\tau_j)}(\delta) \) on \([D, x_j - x_{j-1} - \kappa] \). Hence \( x_j - x_{j-1} - \kappa < \sigma \) for sufficiently large \( n \) and furthermore a.a.s. there exists a sequence \( \kappa(n) = o(1) \) such that \( x_j - x_{j-1} - \kappa(n) < \sigma(n) \).

Now for \( t = \lfloor nx_{j-1} \rfloor + 1, \ldots, \lfloor nx_j \rfloor \), let \( x = t/n \) and \( x' = x - x_{j-1} - \kappa(n) \). By part (v) of Properties 3.1, (4.11), (4.13), and the Lipschitz property of \( \hat{y}_i \), a.a.s. we have

\[
Y_i(t) = nz_i^{(j)}(x' - \epsilon_j) + o(n) = n\hat{y}_i(x - \kappa(n) - \epsilon_j) + o(n) = n\hat{y}_i(t/n) + o(n).
\]

Moreover, the convergence implied by the \( o(n) \) term is uniform over \( t \). This completes the proof of Theorem 4.8.

### 4.2 Changing phase

We usually satisfy the hypotheses of Theorem 4.8 using values calculated numerically and the following theoretical results. The theoretical results show that certain functions are zero at the change of phase (which cannot be checked numerically). They also allow us to specify alternative hypotheses that are easier to satisfy and are sufficient in many cases. We present the results without proof. The proofs are similar to those of Lemma 3.11 and Lemma 3.12, and can be found in the author’s PhD thesis [7, Section 5.5].

First we consider \( \Delta_t^{(\tau_j)} \hat{y}_{\tau_j} \) at \( x_{j-1} \) when \( \tau_j > \tau_{j-1} \). Recall from Definition 3.8 the Independent Types Property.
Lemma 4.9 ([7, Lemma 5.5.1]). Assume that \( j > 1 \) and \( \tau_j > \tau_{j-1} \). Let \( E \) be the clutch matrix for a \( \{\tau_1, \ldots, k\} \)-clutch. If \( \det E(x_{j-1}) \neq 0 \) or the Independent Types Property holds for a phase of type \( \tau_j \), then \( \Delta^{(\tau_j)} y_{\tau_j}(x_{j-1}) = 0 \).

The next lemma gives sufficient conditions for \( q_b^{(\tau_j)} \) to be zero.

Lemma 4.10 ([7, Lemma 5.5.2]). Let \( M^{(\tau_j)} = \{w_1, \ldots, w_a\} \) with \( w_1 < \cdots < w_a \), and let \( x \in [x_{j-1}, x_j] \). If \( \tau \) is such that \( 0 \leq \tau < \tau_j \), \( \tau_j \in M^{(\tau_j)} \), and \( \hat{y}_i(x) = 0 \) for \( i = \tau + 1, \ldots, \tau_j \), then \( q_b^{(\tau_j)}(x) = 0 \) whenever \( w_b \in M^{(\tau_j)} \setminus M^{(\tau)} \).

The above lemma is most useful at the beginning of phase \( j \) when \( \tau_j > \tau_{j-1} \), and at the end of phase \( j - 1 \) when \( \tau_j < \tau_{j-1} \). These cases are given in the next corollary. We may also apply Lemma 4.10 more generally, for example, to the first phase of an algorithm.

Corollary 4.11 ([7, Corollary 5.5.3]). Let \( M^{(\tau_j)} = \{w_1, \ldots, w_a\} \) with \( w_1 < \cdots < w_a \).

(i) Let \( j > 1 \) (so that \( M^{(\tau_{j-1})} \) is defined) and assume that \( \tau_j > \tau_{j-1} \). Then for each \( w_b \in M^{(\tau_j)} \setminus M^{(\tau_{j-1})} \) we have \( q_b^{(\tau_j)}(x_{j-1}) = 0 \).

(ii) Let \( j < K \) (so that \( M^{(\tau_{j+1})} \) is defined) and assume that \( \tau_{j+1} < \tau_j \). If we have \( \tau_j \in M^{(\tau_{j+1})} \), then \( q_b^{(\tau_j)}(x_j) = 0 \) for each \( w_b \in M^{(\tau_j)} \setminus M^{(\tau_{j+1})} \).

When \( q_b^{(\tau_j)}(x_{j-1}) = 0 \), we must consider the derivatives of \( q_b^{(\tau_j)} \) at \( x_{j-1} \). The next lemma shows that \( (\Delta^{(\tau_j)} q_b^{(\tau_j)})(x_{j-1}) \) may also be zero.

Lemma 4.12 ([7, Lemma 5.5.4]). Let \( M^{(\tau_j)} = \{w_1, \ldots, w_a\} \) with \( w_1 < \cdots < w_a \). Assume that \( j > 1 \) and \( \tau_j > \tau_{j-1} \). If the clutch matrix for a \( \{\tau_j, \ldots, k\} \)-clutch has a non-zero determinant or the Independent Types Property holds for a phase of type \( \tau_j \), then \( \Delta^{(\tau_j)} q_b^{(\tau_j)}(x_{j-1}) = 0 \) for \( w_b \in M^{(\tau_j)} \setminus M^{(\tau_{j-1})} \).

Similar to Lemma 4.10, the next lemma considers the derivatives of the functions \( \hat{y}_i \) during a preprocessing subphase.

Lemma 4.13 ([7, Lemma 5.5.5]). Assume that the Independent Types Property holds for a phase of type \( \tau_j \). If \( \tau \in \{0, \ldots, \tau_j - 1\} \) is such that \( \hat{y}_{\tau+1}(x_{j-1}) = \cdots = \hat{y}_{\tau_j}(x_{j-1}) = 0 \), then for all \( i \in M^{(\tau_j)} \setminus M^{(\tau)} \) we have \( \Delta^{(\tau_i)} \hat{y}_i(x_{j-1}) = 0 \).

As with Lemma 4.10 we usually apply Lemma 4.13 with \( \tau = \tau_{j-1} \) when changing to a phase of higher type. Finally we consider the functions \( E_b^{(\tau_j)} \) at a change of phase.

Lemma 4.14 ([7, Lemma 5.5.6]). Assume that \( j > 1 \), so \( M^{(\tau_{j-1})} \) is defined. If the Independent Types Property holds for a phase of type \( \tau_j \), then

\[
E_b^{(\tau_j)}(x_{j-1}) > 0 \quad \text{for} \quad b = \max \{2, 2 + \tau_{j-1} - \tau_j\}, \ldots, |M^{(\tau_j)}|.
\]

In particular, if \( \tau_j > \tau_{j-1} \), then hypothesis (E) is satisfied for phase \( j \).
The hypotheses (A)–(F) given earlier are more general than has been required for the applications considered so far. So we now give two alternative sets of hypotheses: one for changing to a phase of higher type and one for changing to a phase of lower type. These hypotheses assume that the algorithm is changing phase and so cannot be used for the first phase. However, for the first phase, we often know the initial conditions exactly, so it is not hard to check the hypotheses (A)–(F). First we consider changing to a phase of a higher type.

Hypotheses for changing to a phase of higher type

Let \( j > 1 \) be the phase number.

(A2) The phase type \( \tau_j \) is defined (according to Definition 3.13) and \( \tau_j > \tau_{j-1} \).

(B2) At \( x_{j-1} \) we have

1. \((-1)^{|\mathcal{M}(\tau_j)|+1} \det \mathcal{C}(\tau_j) > \delta \),
2. \( (\Delta^{(\tau_j)})^2 \hat{y}_{\tau_j}(x_{j-1}) > 0 \), and
3. \( \Delta(\tau_j) \hat{y}_{\tau_j}(x_{j-1}) \neq 0 \).

(C2) For \( w \in \mathcal{M}(\tau_j) \setminus \mathcal{M}(\tau_{j-1}) \) we have

1. \( \Delta^{(\tau_j)} q_{b_{\tau_j}}(x_{j-1}) > 0 \),
2. \( \Delta^{(\tau_j)} \Delta^{(\tau_j)} q_{b_{\tau_j}}(x_{j-1}) \neq 0 \),
3. \( (\Delta^{(\tau_j)})^2 q_{b_{\tau_j}}(x_{j-1}) > 0 \), and
4. \( (\Delta^{(\tau_j)})^2 \hat{y}_w(x_{j-1}) > 0 \)

where \( b = \text{ix}(w, \mathcal{M}(\tau_j)) \).

(D2) For \( w \in \mathcal{M}(\tau_j) \cap \mathcal{M}(\tau_{j-1}) \) we have \( q_{b_{\tau_j}}(x_{j-1}) > 0 \) and \( \Delta^{(\tau_j)} \hat{y}_w(x_{j-1}) > 0 \) where \( b = \text{ix}(w, \mathcal{M}(\tau_j)) \).

Hypotheses (A2)–(D2) imply the original hypotheses (A)–(F) when the Independent Types Property holds.

Lemma 4.15 ([7, Lemma 5.5.7]). Assume that \( j > 1 \) and the Independent Types Property holds for a phase of type \( \tau_j \). If (A2)–(D2) hold, then there is a phase \( j \) of type \( \tau_j \) and hypotheses (A)–(F) are satisfied for phase \( j \).

Hypotheses for changing to a phase of lower type

Let \( j > 1 \) be the phase number.

(A3) The phase type \( \tau_j \) is defined (according to Definition 3.13) and \( \tau_j < \tau_{j-1} \).

(B3) At \( x_{j-1} \) we have

1. \((-1)^{|\mathcal{M}(\tau_j)|+1} \det \mathcal{C}(\tau_j) > \delta \), and
2. \( E_{b_{\tau_j}}^{(\tau_j)}(x_{j-1}) > 0 \) for \( b = 2, \ldots, \tau_{j-1} - \tau_j + 1 \).
(C3) For \( b = 1, \ldots, |\mathcal{M}^{(\tau)}| \) we have \( q_b^{(\tau)}(x_{j-1}) > 0 \).

(D3) For \( w \in \mathcal{M}^{(\tau)} \setminus \{\tau_j\} \) we have \( \Delta^{(w)}y_w(x_{j-1}) > 0 \).

Again, provided the Independent Types Property holds, hypotheses (A3)–(D3) imply the original hypotheses (A)–(F).

**Lemma 4.16** ([7, Lemma 5.5.8]). Assume that \( j > 1 \) and that the Independent Types Property holds for a phase of type \( \tau_j \). If (A3)–(D3) hold, then there is a phase \( j \) of type \( \tau_j \) and hypotheses (A)–(F) are satisfied for phase \( j \).

Note that Lemma 4.15 and Lemma 4.16 follow easily from the results of the previous section, Lemma 3.16, and Corollary 3.14.

5 Analysing PathDomSetD

We now apply the theory of the previous two sections to the algorithm PathDomSetP, given in Section 2. We start by determining functions that approximate the expected change in the random variables due to an operation.

Recall that \( Z(i,j) = |V_{(i,j)}| \) counts the number of vertices of degree pair \((i,j)\). We also define \( Z = |D| \) to be the number of vertices included in the path dominating set. The operations are defined using free points (either in or out) selected uniformly at random. Let \( P_{\text{in}}(w \in V_{(i,j)}) \) be the probability that a vertex \( w \), selected via a free in-point chosen uniformly at random, has degree pair \((i,j)\). Define \( P_{\text{out}}(w \in V_{(i,j)}) \) similarly. Then

\[
P_{\text{in}}(w \in V_{(i,j)}) = \frac{(d-i)Z(i,j)}{\rho} \quad \text{and} \quad P_{\text{out}}(w \in V_{(i,j)}) = \frac{(d-j)Z(i,j)}{\rho},
\]

where

\[
\rho = \sum_{0 \leq i,j \leq d} (d-i)Z(i,j) = \sum_{0 \leq i,j \leq d} (d-j)Z(i,j).
\]

Since \( \rho = \Omega(n) \) (while the deprioritised algorithm is analysed), the value of \( Z(i,j)/\rho \) during an operation is within \( o(1) \) of its value at the start of the operation. So we may treat each \( Z(i,j) \) as fixed throughout each operation.

Now consider an operation of PathDomSetP. During an operation there are five sorts of vertices:

- the vertex \( u \) processed by the operation,
- vertices that have no associated free points exposed,
- vertices added to \( D \) (which are either an in-neighbour or an out-neighbour of \( u \)),
- vertices that are chosen via a randomly selected in-point, called \( \text{in-incs} \), and
- vertices that are chosen via a randomly selected out-point, called \( \text{out-incs} \).

For convenience we extend the definition of \( Z(i,j) \) so that \( Z(i,j) = 0 \) for \( i < 0 \) or \( j < 0 \). Then the expected change in \( Z(i,j) \) due to an in-inc \( w \) is \( \text{In}_{(i,j)} = o(1) \) where

\[
\text{In}_{(i,j)} = P_{\text{in}}(w \in V_{(i-1,j)}) - P_{\text{in}}(w \in V_{(i,j)}) = ((d+1-i)Z(i-1,j) - (d-i)Z(i,j))/\rho.
\]
Similarly, the expected change in $Z_{(i,j)}$ due to an out-inc $w$ is $\Out_{(i,j)} + o(1)$ where
\[
\Out_{(i,j)} = \Pr(\text{out}(w \in V_{(i,j-1)}) - \Pr(\text{out}(w \in V_{(i,j)})) = \frac{(d + 1 - j)Z_{(i,j-1)} - (d - j)Z_{(i,j)})}{\rho}.
\]

Now consider an operation that processes a vertex $u$ of degree pair $(p, q)$. If $p = 0$, then we expose a free in-point associated with $u$ to obtain a new in-neighbour $w_1$. So the expected change in $Z_{(i,j)}$ due to exposing the free points associated with $w_1$ is $\Psi_{(i,j)}^{\text{in}} + o(1)$ where
\[
\Psi_{(i,j)}^{\text{in}} = \delta_{i,d} \delta_{j,d} - \Pr_{\text{in}}(w_1 \in V_{(i,j)}) + \left[ \sum_{r=0}^{d} \sum_{s=0}^{d} (d - r)\Pr_{\text{out}}(w_1 \in V_{(r,s)}) \right] \Out_{(i,j)}
\]
\[
+ \left[ \sum_{r=0}^{d} \sum_{s=0}^{d} (d - s - 1)\Pr_{\text{out}}(w_1 \in V_{(r,s)}) \right] \In_{(i,j)}.
\]

Similarly, if $q = 0$, then we expose an out-point associated with $u$ to obtain a new out-neighbour $w_2$. The expected change in $Z_{(i,j)}$ due to exposing the free points associated with $w_2$ is $\Psi_{(i,j)}^{\text{out}} + o(1)$ where
\[
\Psi_{(i,j)}^{\text{out}} = \delta_{i,d} \delta_{j,d} - \Pr_{\text{in}}(w_2 \in V_{(i,j)}) + \left[ \sum_{r=0}^{d} \sum_{s=0}^{d} (d - r - 1)\Pr_{\text{in}}(w_2 \in V_{(r,s)}) \right] \Out_{(i,j)}
\]
\[
+ \left[ \sum_{r=0}^{d} \sum_{s=0}^{d} (d - s)\Pr_{\text{in}}(w_2 \in V_{(r,s)}) \right] \In_{(i,j)}.
\]

The change in $Z_{(i,j)}$ from exposing the free points associated with the processed vertex $u$ is
\[
\begin{cases}
\delta_{i,1}\delta_{j,1} - \delta_{i,0}\delta_{j,0} & \text{if } p = q = 0, \\
\delta_{i,1}\delta_{j,q} - \delta_{i,0}\delta_{j,q} & \text{if } p = 0 \text{ and } q > 0, \\
\delta_{i,p}\delta_{j,1} - \delta_{i,p}\delta_{j,0} & \text{if } p > 0 \text{ and } q = 0.
\end{cases}
\]

Therefore the expected change in $Z_{(i,j)}$ due to an operation of type $r$ is $\Opr_{(i,j)} + o(1)$ where
\[
\Opr_{(i,j)} = \begin{cases}
\delta_{i,1}\delta_{j,1} - \delta_{i,0}\delta_{j,0} + \Psi_{(i,j)}^{\text{in}} + \Psi_{(i,j)}^{\text{out}} & \text{if } p = q = 0, \\
\delta_{i,1}\delta_{j,q} - \delta_{i,0}\delta_{j,q} + \Psi_{(i,j)}^{\text{in}} & \text{if } p = 0 \text{ and } q > 0, \\
\delta_{i,p}\delta_{j,1} - \delta_{i,p}\delta_{j,0} + \Psi_{(i,j)}^{\text{out}} & \text{if } p > 0 \text{ and } q = 0.
\end{cases}
\]

Also, the expected change in $Z$ due to an operation of type $r$ is $\text{dom}_r + o(1)$ where $\text{dom}_r = \delta_{p,0} + \delta_{q,0}$.

Now we set $Z_{(i,j)}(t) = nz_{(i,j)}(t/n)$ and $Z(t) = nz(t/n)$ and write $\Opr_{(i,j)}$ and $\text{dom}_r$ in terms of $z_{(i,j)}$ and $z$ to obtain functions $f_{(i,j)}^{(r)}$ and $f^{(r)}$ respectively. The functions $f_{(i,j)}^{(r)}$ and $f^{(r)}$ show that part (iii) of Properties 3.1 is satisfied, with $H = \rho/n$ (written in terms of $z_{(i,j)}$). From the definition of PathDomSetP it is clear that the remaining parts are also satisfied. Thus we are able to define the deprioritised algorithm PathDomSetD based on PathDomSetP.

We now determine the irreducible type sets using the DPPD method. Recall the definition of $B_{(i,j)}$-determined and the functions $\text{dp}$ and $\nu$ from Section 3.1.3. For $(i,j)$ corresponding to an operation type, the function $f_{(i,j)}^{(r)}$ is $B_{(i,j)}$-determined for a phase of type $\tau$ if
(a) every term in the numerator of $f_{(i,j)}^{(r)} + \delta_{\nu(i,j),r}$ contains at least one variable in the set $\{z_{(i,j)}, z_{(i-1,j)}, z_{(i,j-1)}\}$, and

(b) for all $(p, q) \in \{(i - 1, j), (i, j - 1)\}$, if $p, q \geq 0$, then there exists a term of the numerator of $f_{(i,j)}^{(r)} + \delta_{\nu(i,j),r}$ containing the variable $z_{(p,q)}$ but no variables in the set $\{z_{dp(j)} : j = \tau + 1, \ldots, k\}$.

Now fix $(i, j)$ with $i = 0$ or $j = 0$. Notice that every term of the numerators of $I_{(i,j)}$ and $Out_{(i,j)}$ involves $z_{(i,j)}$ or $z_{(i-1,j)}$ or $z_{(i,j-1)}$. Therefore every term of the numerator of $\Psi_{(i,j)}^{in}$, $\Phi_{(i,j)}^{in}$, and $f_{(i,j)}^{(r)} + \delta_{\nu(i,j),r}$ involves $z_{(i,j)}$ or $z_{(i-1,j)}$ or $z_{(i,j-1)}$. So part (a) above is satisfied. It is also clear from the definition of $f_{(i,j)}^{(r)} + \delta_{\nu(i,j),r}$ that for $(p, q) \in B_{(i,j)}$, the numerator of $f_{(i,j)}^{(r)} + \delta_{\nu(i,j),r}$ contains a term involving only the variables $z_{(p,q)}$ and $z_{(0,0)}$. Hence part (b) above is also satisfied. So applying Lemma 3.10 we determine the irreducible type sets and show that the Independent Types Property property holds for all phase types.

## 5.1 Numerical analysis

We now use Theorem 4.8 to analyse PathDomSetD for a given $d$. As the Independent Types Property holds for all phase types, we are able to use the alternative hypotheses (A2)–(D2) and (A3)–(D3) for all phases except phase one. These hypotheses are checked using numerical approximations to the functions $\hat{y}_i$ defined by $z_{(p,q)} = \hat{y}_{\nu(p,q)}$ and $z = \hat{y}_{36}$. The numerical approximations are obtained by solving the corresponding differential equations (non-rigorously) using the fourth order Runge-Kutta method. The solutions are extended as far as possible while the hypotheses are satisfied for some $\delta > 0$. We then take $\delta$ sufficiently small so as to include each point of the solutions.

From the analysis of PathDomSetD we obtain a.a.s. bounds on the size of the path dominating set returned by PathDomSetD. The upper bounds are also a.a.s. upper bounds on the minimum size of a path dominating set of a random $d$-in $d$-out digraph. The bounds for $d = 2, 3, 4, 5$ are given in Table 5.1.

We now present the analysis for $d = 5$. Earlier we showed that PathDomSetP satisfies Properties 3.1. Note that $\hat{y}_i(0) = \lim_{n \to \infty} Y_i(0)/n = \delta_{i,0}$. It is easily checked that $(0, \hat{y}_0(0), \ldots, \hat{y}_{36}(0)) \in D(\delta)$. So it just remains to show that hypotheses (A)–(F) hold for each phase.

We expect most of the early operations of PathDomSetP to have type 2. So we start the analysis with a phase of type 2. Since the initial conditions are known exactly, the values of the functions for the first phase are calculated exactly. We present the function values required to check hypotheses (A)–(F) in Table 5.2.

Computing numerical solutions to the differential equations we find that $\hat{y}_{36}(x_1)$ satisfies $0.0487158 < \hat{y}_{36}(x_1) < 0.0487159$. The solutions exit $V(2)(\delta)$ at the boundary $q_1^{(2)} = E_2^{(2)} = 0$ (note that these functions are equal by definition). Thus there is a next phase and it has type 3. Let $\hat{x}_1$ be such that $\hat{y}_{36}(\hat{x}_1) = 0.0487158$. We check the hypotheses for phase two at $\hat{x}_1$ instead of at $x_1$. The data for the remaining phases is given in Tables 5.3 to 5.6. We see that when the analysis finishes, after phase five, we have $\hat{y}_{36}(x_5) < 0.3926120$. Thus $0.3927n$ is an a.a.s. upper bound on the minimum size of a path dominating set of a random 5-in 5-out digraph.
Table 5.1: Asymptotically almost sure upper bounds on the minimum size of a path dominating set of a random $d$-in $d$-out digraph.

<table>
<thead>
<tr>
<th>$d$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>upper bound</td>
<td>$0.5529n$</td>
<td>$0.4892n$</td>
<td>$0.4354n$</td>
<td>$0.3927n$</td>
</tr>
</tbody>
</table>

Phase One of type 2

Start of Phase

Hypothesis (A) $\det C^{(2)} = 1$, $\hat{y}_2 = 0$, $q_b^{(2)} = \delta_{b,1}$ for $b = 1, 2, 3$,

$E_b^{(2)} = 1$ for $b = 2, 3$

Hypothesis (B) $\Delta^{(2)} \hat{y}_2(x_0) = 4$, $\Delta^{(2)} q_2^{(2)}(x_0) = 12.8$, $\Delta^{(2)} q_3^{(2)}(x_0) = 16$

Hypothesis (C) $\Delta^{(r)} \hat{y}_2(x_0) = 9$, $\Delta^{(r)} \hat{y}_i(x_0) = 0$ for $i = 3, 4$,

$(\Delta^{(r)})^2 \hat{y}_i(x_0) = 64.8$ for $i = 3, 4$

Hypothesis (D) $\Delta^{(r)} q_2^{(2)}(x_0) = 28.8$, $\Delta^{(r)} q_3^{(2)}(x_0) = 36$

Hypothesis (E) See values for hypothesis (A)

Hypothesis (F) See values above

End of Phase $q_1^{(2)} = E_2^{(2)} = 0$, $0.0487158 < \hat{y}_{36}(x_1) < 0.0487159$

Table 5.2: Function values for checking hypotheses (A)–(F) for phase one.

Phase Two of type 3

Start of Phase

Hypothesis (A2) From above

Hypothesis (B2) $(-1)^{|M^{(3)}|+1} \det C^{(3)}(\hat{x}_1) = 0.88769...$, $(\Delta^{(3)})^2 \hat{y}_3(\hat{x}_1) = 13.32482...$

$\Delta^{(r)} \Delta^{(3)} \hat{y}_3(\hat{x}_1) = 12.50778...$

Hypothesis (C2) $\Delta^{(r)} q_3^{(3)}(\hat{x}_1) = 2.53608...$, $\Delta^{(r)} \Delta^{(3)} q_3^{(3)}(\hat{x}_1) = 31.58252...$

$(\Delta^{(3)})^2 q_3^{(3)}(\hat{x}_1) = 33.64556...$, $(\Delta^{(r)})^2 \hat{y}_5(\hat{x}_1) = 5.71065...$

Hypothesis (D2) $q_1^{(3)}(\hat{x}_1) = 0.449604...$, $q_2^{(3)}(\hat{x}_1) = 0.44165...$

$\Delta^{(r)} \hat{y}_3(\hat{x}_1) = 1.00437...$, $\Delta^{(r)} \hat{y}_4(\hat{x}_1) = 0.99561...$

End of Phase $\hat{y}_3 = q_3^{(3)} = 0$, $0.2824230 < \hat{y}_{36}(x_2) < 0.2824231$

Table 5.3: Function values for checking hypotheses (A2)–(D2) for phase two.
Phase Three of type 2

<table>
<thead>
<tr>
<th>Start of Phase</th>
<th>( \hat{y}_{36}(\hat{x}_2) = 0.2824230 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypothesis (A3)</td>
<td>Checked at end of phase two</td>
</tr>
<tr>
<td>Hypothesis (B3)</td>
<td>((-1)^{</td>
</tr>
<tr>
<td>Hypothesis (C3)</td>
<td>(q_1^{(2)}(\hat{x}_2) = 0.36979...,) (q_2^{(2)}(\hat{x}_2) = 0.25661...,) (q_3^{(2)}(\hat{x}_2) = 0.29629...)</td>
</tr>
<tr>
<td>Hypothesis (D3)</td>
<td>(\Delta^{(r)}\hat{y}_3(\hat{x}_2) = 0.59053...,) (\Delta^{(r)}\hat{y}_4(\hat{x}_2) = 0.60693...)</td>
</tr>
<tr>
<td>End of Phase</td>
<td>(\hat{y}<em>2 = q_3^{(2)} = 0,) (0.3509639 &lt; \hat{y}</em>{36}(x_3) &lt; 0.3509640)</td>
</tr>
</tbody>
</table>

Table 5.4: Function values for checking hypotheses (A3)–(D3) for phase three.

Phase Four of type 1

<table>
<thead>
<tr>
<th>Start of Phase</th>
<th>( \hat{y}_{36}(\hat{x}_3) = 0.3509639 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypothesis (A3)</td>
<td>Checked at end of phase three</td>
</tr>
<tr>
<td>Hypothesis (B3)</td>
<td>((-1)^{</td>
</tr>
<tr>
<td>Hypothesis (C3)</td>
<td>(q_1^{(1)}(\hat{x}_3) = 0.74298...,) (q_2^{(1)}(\hat{x}_3) = 0.06870...,) (q_3^{(1)}(\hat{x}_3) = 0.16535...)</td>
</tr>
<tr>
<td>Hypothesis (D3)</td>
<td>(\Delta^{(r)}\hat{y}_2(\hat{x}_3) = 0.16035...,) (\Delta^{(r)}\hat{y}_3(\hat{x}_3) = 0.30375...)</td>
</tr>
<tr>
<td>End of Phase</td>
<td>(\hat{y}<em>1 = q_2^{(1)} = 0,) (0.3789442 &lt; \hat{y}</em>{36}(x_4) &lt; 0.3789443)</td>
</tr>
</tbody>
</table>

Table 5.5: Function values for checking hypotheses (A3)–(D3) for phase four.

Phase Five of type 0

<table>
<thead>
<tr>
<th>Start of Phase</th>
<th>( \hat{y}_{36}(\hat{x}_4) = 0.3789442 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypothesis (A3)</td>
<td>Checked at end of phase four</td>
</tr>
<tr>
<td>Hypothesis (B3)</td>
<td>((-1)^{</td>
</tr>
<tr>
<td>Hypothesis (C3)</td>
<td>(q_1^{(0)}(\hat{x}_4) = 0.88161...,) (q_2^{(0)}(\hat{x}_4) = 0.10362...,) (q_3^{(0)}(\hat{x}_4) = 0.10347...)</td>
</tr>
<tr>
<td>Hypothesis (D3)</td>
<td>(\Delta^{(r)}\hat{y}_1(\hat{x}_4) = 0.10518...,) (\Delta^{(r)}\hat{y}_2(\hat{x}_4) = 0.10503...)</td>
</tr>
<tr>
<td>End of Phase</td>
<td>(\hat{y}<em>0 = q_2^{(0)} = q_3^{(0)} = 0,) (0.3926119 &lt; \hat{y}</em>{36}(x_5) &lt; 0.3926120)</td>
</tr>
</tbody>
</table>

Table 5.6: Function values for checking hypotheses (A3)–(D3) for phase five.
References


