On Boundedness in Depth in the π-Calculus*

Roland Meyer

University of Oldenburg
Roland.Meyer@informatik.uni-oldenburg.de

Abstract. We investigate the class $\mathcal{P}_{BD}$ of π-Calculus processes that are bounded in the function *depth*. First, we show that boundedness in depth has an intuitive characterisation when we understand processes as graphs: a process is bounded in depth if and only if the length of the simple paths is bounded. The proof is based on a new normal form for the π-Calculus called *anchored fragments*. Using this concept, we then show that processes of bounded depth have well-structured transition systems (WSTS). As a consequence, the termination problem is decidable for this class of processes. The instantiation of the WSTS framework employs a new well-quasi-ordering for processes in $\mathcal{P}_{BD}$.

1 Introduction

Concurrent systems are known to be hard to design correctly. Dynamically reconfigurable systems add to concurrency the problem of changing connection structures between system components. To ensure the correct behaviour of systems, automatic verification techniques have proven useful. This automation comes with a tradeoff. To automate the analysis requires a decidable class of models, but to model the systems of interest requires an expressive class. We use the π-Calculus to model dynamically reconfigurable systems [17, 18]. The contribution of this paper is the up-to-now most expressive subclass of π-Calculus for which termination is decidable. The importance of termination for the π-Calculus has been recognised in [19, 5].

The class $\mathcal{P}_{BD}$ we propose contains the processes that are *bounded in depth*. The function *depth* measures the interdependence of restricted names in process terms. Boundedness in depth is a very liberal requirement as it turns out that all decidable subclasses of π-Calculus known so far are subclasses of $\mathcal{P}_{BD}$: finitary agents [9], finite control processes [4], bounded processes [3], unique receiver and bounded input systems (up to bisimilarity) [2], finite handler processes [14], structurally stationary processes [14], and restriction-free processes [2].

But the definition of depth is difficult to grasp as the function refers to all processes in a congruence class. To provide an intuition to $\mathcal{P}_{BD}$, we make use of the standard graph-theoretic interpretation of the π-Calculus [17, 18]. Our first main result states that boundedness in depth is equivalent to boundedness in the

* This work was supported by the German Research Council (DFG) as part of the Graduate School "TrustSoft" (GRK 1076/1).

length of the simple paths (i.e., without repetition of edges) in the graphs. The proof is based on a new normal form for processes called *anchored fragments*.

The decidability result for $\mathcal{P}_{BD}$ is obtained by viewing this class as an instance of *well-structured transition systems (WSTS)* [10, 1, 11]. WSTS are a framework for infinite state systems that generalises decidability results for particular models. Technically, a WSTS is a transition system with an ordering relation on the states which is compatible with the transition relation. Depending on the ordering, the compatibility, and decidability properties the framework yields decision procedures, e.g., for termination [10, 11] or simulation [1].

Our second main result is the instantiation of the WSTS framework for processes of bounded depth. As a consequence, we inherit the decision procedure for termination in [10, 11]. The technical contribution is a new ordering $\preceq_{\mathcal{P}_{BD}}$ on processes which we show to be a *well-quasi-ordering (wqo)* (i.e., in every infinite sequence of processes two comparable processes can be found) for processes of bounded depth. In the proof, anchored fragments again play a vital role. Since the ordering $\preceq_{\mathcal{P}_{BD}}$ is a simulation relation it is compatible with the reaction relation of the π-Calculus in a strong sense.

2 Preliminaries

The π-Calculus We use a π-Calculus with parameterised recursion as proposed in [18]. Let the set $(a, b \in) \mathcal{N}$ of *names* contain the channels and messages that occur in communications. A process consumes *prefixes* π to communicate with other processes or to perform silent actions. The prefixes are

$$\pi ::= \overline{a}\langle b\rangle \mid a(x) \mid \tau.$$

The *output action* $\overline{a}\langle b\rangle$ sends the name b on channel a. The *input action* $a(x)$ receives a name that replaces x on a. The τ symbol stands for a *silent action*.

To denote recursive processes we use *process identifiers* K, each defined by an equation $K(\tilde{x}) := P$. When the identifier is *called*, $K\lfloor\tilde{a}\rfloor$, it is replaced by the process P where the names $\tilde{x}$ are replaced by $\tilde{a}$. More precisely, a *substitution* $\sigma = \{\tilde{a}/\tilde{x}\}$ is a function that maps the names in $\tilde{x}$ to $\tilde{a}$ and is the identity on all names not in $\tilde{x}$. The *application of a substitution*, $P\{\tilde{a}/\tilde{x}\}$, is defined in the standard way [18]. A π-Calculus process is a call to an identifier, $K\lfloor\tilde{a}\rfloor$, a *choice process* deciding between prefixes, $\Sigma_{i\in I}\pi_i.P_i$, a *parallel composition* of processes, $P_1 \mid P_2$, or the *restriction* of a name in a process, $\nu a.P$:

$$P ::= K\lfloor\tilde{a}\rfloor \mid \Sigma_{i\in I}\pi_i.P_i \mid P_1|P_2 \mid \nu a.P.$$

The *set of all processes* is $\mathcal{P}$. We abbreviate empty sums (with $I = \emptyset$) by $\mathbf{0}$ and arbitrary sums by M or N. By $\Pi_{i\in I}P_i$ we denote the parallel composition of several processes P_i with $i \in I$. The processes $K\lfloor\tilde{a}\rfloor$ and $\Sigma_{i\in I\neq\emptyset}\pi_i.P_i$ are called *sequential*. By $\mathcal{S}(P)$ we refer to the *set of sequential processes* in P. The function

is defined inductively by $\mathcal{S}(\mathbf{0}) := \emptyset$, $\mathcal{S}(K\lfloor\tilde{a}\rfloor) := \{K\lfloor\tilde{a}\rfloor\}$, $\mathcal{S}(\Sigma_{i\in I\neq\emptyset}\pi_i.P_i) := \{\Sigma_{i\in I\neq\emptyset}\pi_i.P_i\}$, $\mathcal{S}(P_1 \mid P_2) := \mathcal{S}(P_1) \cup \mathcal{S}(P_2)$, and $\mathcal{S}(\nu a.P) := \mathcal{S}(P)$.

The input action $a(b)$ and the restriction $\nu c.P$ *bind* b and c, respectively. The *set of bound names* in P is $bn\,(P)$. If we refer to the *set of restricted names* in P, $rn\,(P) \subseteq bn\,(P)$, we mean the restricted names that are not covered by prefixes. A name which occurs not bound in P is *free* and the *set of free names* in P is $fn\,(P)$. We permit α-conversion of bound names. Therefore, wlog. we assume $bn\,(P) \cap fn\,(P) = \emptyset$. Unless otherwise stated, we assume that a name is bound at most once in a process. In a defining equation $K(\tilde{x}) := P$ we require $fn\,(P) \subseteq \tilde{x}$. If a substitution $\{\tilde{a}/\tilde{x}\}$ is applied to a process P, we assume $bn\,(P) \cap (\tilde{a} \cup \tilde{x}) = \emptyset$.

The results achieved in this paper make heavy use of the *structural congruence* relation $\equiv$ of processes. It is the smallest congruence where α-conversion of bound names is allowed, $+$ and $\mid$ are commutative and associative and have $\mathbf{0}$ as neutral element, and the following laws for restriction hold:

$$\nu x.\nu y.P \equiv \nu y.\nu x.P \qquad\qquad \nu x.\mathbf{0} \equiv \mathbf{0}$$
$$\nu x.(P \mid Q) \equiv P \mid (\nu x.Q), \text{ if } x \notin fn\,(P).$$

The latter law is called *scope extrusion*.

We distinguish two normal forms for processes. A process $\nu\tilde{a}.(P_1 \mid \ldots \mid P_n)$ where $\tilde{a} \subseteq fn\,(P_1 \mid \ldots \mid P_n)$ and all P_i are sequential is in *standard form* [17]. Via structural congruence every process P can be rewritten as a process P_{sf} in standard form as follows. First, the scope of every restricted name not under a prefix is extruded over all processes composed in parallel. Then unused restricted names and empty sums are removed. Since all bound names are different and disjoint with the free names, α-conversion is not required. Thus, the rewriting does not change the sequential processes, $\mathcal{S}(P) = \mathcal{S}(P_{sf})$.

The *restricted form* [14] is based on the notion of *fragments*, i.e., processes where the scopes of restricted names are minimal:

$$F ::= K\lfloor\tilde{a}\rfloor \mid \Sigma_{i\in I\neq\emptyset}\pi_i.P_i \mid \nu a.(F_1 \mid \ldots \mid F_n),$$

where $a \in fn\,(F_i)$ for all i. The *set of all fragments* is $(F, G \in)\ \mathcal{P}_{\mathcal{F}}$. Fragments that are sequential processes, $K\lfloor\tilde{a}\rfloor$ or $\Sigma_{i\in I\neq\emptyset}\pi_i.P_i$, are *elementary* and referred to by F_e. A process P_ν is in *restricted form*, if it is a parallel composition of fragments, $P_\nu = \Pi_{i\in I}F_i$. The *set of fragments in* P_ν is $Frag\,(P_\nu) := \bigcup_{i\in I}\{F_i\}$. The *set of all processes in restricted form* is $\mathcal{P}_\nu$.

To compute the restricted form $P_\nu \in \mathcal{P}_\nu$ of a process $P \in \mathcal{P}$, we minimise the scopes of all restricted names not under a prefix and remove processes congruent with $\mathbf{0}$, in particular unused restricted names. Again, this does not change the sequential processes, $\mathcal{S}(P) = \mathcal{S}(P_\nu)$. The restricted form of a process is invariant under structural congruence up to rewriting of fragments: $P \equiv Q$ iff $P_\nu \widehat{\equiv} Q_\nu$, where $\widehat{\equiv}$ is the smallest equivalence on processes in restricted form that permits

(1) associativity and commutativity with regard to $|$ and (2) replacing fragments by structurally congruent ones, i.e., $F \mid P_\nu \mathrel{\widehat{=}} G \mid P_\nu$ if $F \equiv G$.

The behaviour of π-Calculus processes is determined by the *reaction relation* $\rightarrow \subseteq \mathcal{P} \times \mathcal{P}$ defined by the following rules:

$$\text{(Tau) } \tau.P + M \rightarrow P$$

$$\text{(React) } (x(y).P + M) \mid (\overline{x}\langle z\rangle.Q + N) \rightarrow P\{z/y\} \mid Q$$

$$\text{(Const) } K\lfloor \tilde{a} \rfloor \rightarrow P\{\tilde{a}/\tilde{x}\}, \text{ if } K(\tilde{x}) := P$$

$$\text{(Par) } \frac{P \rightarrow P'}{P \mid Q \rightarrow P' \mid Q} \qquad \text{(Res) } \frac{P \rightarrow P'}{\nu a.P \rightarrow \nu a.P'}$$

$$\text{(Struct) } \frac{P \rightarrow P'}{Q \rightarrow Q'}, \text{ if } P \equiv Q \text{ and } P' \equiv Q'.$$

By $\mathit{Reach}\,(P)$ we denote the *set of all processes reachable from* P with the reaction relation. The reaction relation is *image finite*, i.e., for every process P there are up to structural congruence only finitely many Q with $P \rightarrow Q$.

To relate a reachable fragment $F \in \mathit{Frag}\,(\mathit{Reach}\,(P))$ with the initial process P, we recall that F consists of *derivatives* of P [14]. Derivatives are sequential subprocesses of P gained by removing prefixes as if they were communicated. Let P use $n \in \mathbb{N} = \{0, 1, 2, \ldots\}$ recursive definitions $K_i(\tilde{x}_i) := P_i$. We define $\mathit{derivatives}(P) := \mathit{der}(P) \cup \bigcup_{i=1}^{n} \mathit{der}(P_i)$, where $\mathit{der}(\mathbf{0}) := \emptyset$, $\mathit{der}(K\lfloor \tilde{a} \rfloor) := \{K\lfloor \tilde{a} \rfloor\}$, $\mathit{der}(\Sigma_{i \in I \neq \emptyset} \pi_i.P_i) := \{\Sigma_{i \in I \neq \emptyset} \pi_i.P_i\} \cup \bigcup_{i \in I} \mathit{der}(P_i)$, $\mathit{der}(P_1 | P_2) := \mathit{der}(P_1) \cup \mathit{der}(P_2)$, and $\mathit{der}(\nu a.P) := \mathit{der}(P)$. Then every $F \in \mathit{Frag}\,(\mathit{Reach}\,(P))$ is structurally congruent with $\nu\tilde{a}.(\Pi_{i \in I \neq \emptyset} Q_i\sigma_i)$, where $Q_i \in \mathit{derivatives}(P)$ and $\sigma_i : \mathit{fn}\,(Q_i) \rightarrow \mathit{fn}\,(P) \cup \tilde{a}$.

To define the function *depth*, we require the *nesting of restrictions* measured as follows: $\mathit{nest}_\nu\,(K\lfloor \tilde{a} \rfloor) := 0$, $\mathit{nest}_\nu\,(\Sigma_{i \in I} \pi_i.P_i) := 0$, $\mathit{nest}_\nu\,(P_1 \mid P_2) := \mathit{max}\{\mathit{nest}_\nu\,(P_1), \mathit{nest}_\nu\,(P_2)\}$, and $\mathit{nest}_\nu\,(\nu a.P) := 1 + \mathit{nest}_\nu\,(P)$.

Definition 1. The *depth* of $F \in \mathcal{P}_\mathcal{F}$ is the minimal nesting of restrictions in all fragments in the congruence class: $\mathit{depth}(F) := \mathit{min}\{\mathit{nest}_\nu\,(F') \mid F' \equiv F\}$. A process $P \in \mathcal{P}$ is *bounded in depth*, iff there is $k_D \in \mathbb{N}$ such that the depth of all reachable fragments is less or equal to k_D, i.e.,

$$\exists k_D \in \mathbb{N} : \forall Q \in \mathit{Reach}\,(P) : \forall F \in \mathit{Frag}\,(Q_\nu) : \mathit{depth}(F) \leq k_D.$$

The *set of all processes that are bounded in depth* is $\mathcal{P}_{BD}$. ◊

Well-Quasi-Orderings A *quasi-ordering (qo)* on a set of elements A is a reflexive and transitive relation $\preceq_A \subseteq A \times A$. We also call $(A, \preceq_A)$ a qo. The qo $(A, \preceq_A)$ is a *well-quasi-ordering (wqo)*, iff in every infinite sequence $(a_i)_{i \in \mathbb{N}}$ in A there are two comparable elements, i.e., there are indices $i < j$ with $a_i \preceq_A a_j$.

A result by Higman [13] lifts a wqo $\preceq_A$ on a set of elements A to a wqo $\preceq_A^H$ on the set of finite sequences A^*. The ordering $u \preceq_A^H v$ demands u to be a subsequence of v which is dominated elementwise wrt. $\preceq_A$, i.e., $u = (u_1, \ldots, u_m)$

and $v = (v_1, \ldots, v_n)$ and there are $1 \leq i_1 < \ldots < i_m \leq n$ such that $u_k \preceq_A v_{i_k}$ for all $1 \leq k \leq m$.

In Section 4 we define a qo on fragments. To prove it is a wqo, we relate it with a wqo on trees. Consider a qo $(A, \preceq_A)$. The *trees over* A are defined by

$$T ::= a \mid (a, (T_1, \ldots, T_n)),$$

where $a \in A$. The *set of all trees over* A is $\mathcal{T}(A)$. The *height* of a tree is measured similar to the nesting of restrictions in fragments, $\mathit{height}(a) := 0$ and $\mathit{height}((a, (T_1, \ldots, T_n))) := 1 + \mathit{max}\{\mathit{height}(T_i) \mid 1 \leq i \leq n\}$. For $n \in \mathbb{N}$ we denote by $\mathcal{T}(A)_n$ the trees of height less or equal to n.

We use the *rooted tree embedding* $\preceq_{\mathcal{T}(A)}$ as qo on the trees in $\mathcal{T}(A)$. Intuitively, $T_1 \preceq_{\mathcal{T}(A)} T_2$ if T_1 is a subtree of T_2 so that the levels of T_1 are preserved in T_2. In particular, the root of T_1 is mapped to the root of T_2 and the leaves in T_1 are leaves in T_2. Technically, the rooted tree embedding is defined by two rules. If $a \preceq_A a'$ then $a \preceq_{\mathcal{T}(A)} a'$ (Elem) and if $a \preceq_A a'$ and $(T_1, \ldots, T_m) \preceq^H_{\mathcal{T}(A)} (T'_1, \ldots, T'_n)$ then $(a, (T_1, \ldots, T_m)) \preceq_{\mathcal{T}(A)} (a', (T'_1, \ldots, T'_n))$ (Comp). It is not hard to see that the relation $\preceq_{\mathcal{T}(A)} \subseteq \mathcal{T}(A) \times \mathcal{T}(A)$ is a qo. It is a wqo on trees of bounded height.

Lemma 1. *If $(A, \preceq_A)$ is a wqo then $(\mathcal{T}(A)_n, \preceq_{\mathcal{T}(A)})$ is a wqo for all $n \in \mathbb{N}$.*

3 A Characterisation of Boundedness in Depth

In this section, we interpret fragments F as hypergraphs $\mathcal{G}[\![F]\!]$. With this interpretation we call the process $P \in \mathcal{P}$ *bounded in the simple paths*, iff there is $k_{Sim} \in \mathbb{N}$ such that the length of the longest simple path in the hypergraphs of all reachable fragments is less or equal to k_{Sim}, i.e.,

$$\exists k_{Sim} \in \mathbb{N} : \forall Q \in \mathit{Reach}\,(P) : \forall F \in \mathit{Frag}\,(Q_\nu) : \mathit{lsp}(\mathcal{G}[\![F]\!]) \leq k_{Sim},$$

where $\mathit{lsp}(\mathcal{G}[\![F]\!])$ denotes the length of the longest simple path in $\mathcal{G}[\![F]\!]$. We prove that a process is bounded in depth if and only if it is bounded in the simple paths. Thus, processes in $\mathcal{P}_{BD}$ can be intuitively understood as hypergraphs where the length of the simple paths is bounded.

The main technical contribution is the definition of *anchored fragments*. In this section, we use them to derive boundedness in depth from boundedness in the simple paths (Lemma 3). In Section 4 they help us prove that the given qo is a wqo. In particular we need that the nesting of restrictions in anchored fragments is bounded if the depth is (Corollary 1). Before we turn to anchored fragments, we make the interpretation of processes as hypergraphs precise.

3.1 The Graph-theoretic Interpretation of the π-Calculus

A *hypergraph* [12] is a graph where several vertices may be connected with one hyperedge, i.e., it is a tuple $\mathcal{G} = (V, E, l, inc)$, where V is a finite set of *vertices*, E is a finite set of *hyperedges*, $l : V \to \mathcal{P}$ is a *vertex labelling function*, and $inc : E \to \mathbb{P}(V)$ is an *incidence function*. In the graphical representation we draw a dot labelled by $l(v)$ for each $v \in V$ and a box labelled by e for every $e \in E$. There is an arc between v and e, if $v \in inc(e)$. In our setting edges are names, $E \subseteq \mathcal{N}$. We also call hypergraphs *graphs* and hyperedges *edges*.

Two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ are *equal*, $\mathcal{G}_1 = \mathcal{G}_2$, if $E_1 = E_2$ and there is a bijection $f : V_1 \to V_2$ that is compatible with the labelling and the incidence functions. Hence, the identity of elements $v \in V$ is not important and we can always assume $V_1 \cap V_2 = \emptyset$.

A *path* in $\mathcal{G}$ is a finite sequence $p = (v_1, e_1, \ldots, v_n, e_n, v_{n+1})$ such that the edges e_i connect v_i and v_{i+1}, i.e., $v_i, v_{i+1} \in inc(e_i)$ for all i. The *length* of p, $length(p)$, is the number of edges in p. By $fe(p)$ we refer to the *first element in* p, v_1. A path is *simple*, if $e_i \neq e_j$ for all $i \neq j$. By $lsp(\mathcal{G})$ we denote the *length of the longest simple path* in $\mathcal{G}$. The *set of all paths* in $\mathcal{G}$ is $Paths(\mathcal{G})$.

We require three operations on graphs. The *disjoint union* of $\mathcal{G}_1$ and $\mathcal{G}_2$, where $E_1 \cap E_2 = \emptyset$, puts both graphs side by side. Formally, it is the graph $\mathcal{G}_1 \uplus \mathcal{G}_2 := (V_1 \uplus V_2, E_1 \uplus E_2, l_1 \uplus l_2, inc_1 \uplus inc_2)$. The *connect operator* takes a graph $\mathcal{G}$ and a name $a \notin E$. The result is the graph $\mathcal{G} \otimes a$, where a is added to E. The new edge connects the processes that have a as a free name, i.e., $\mathcal{G} \otimes a := (V, E \uplus \{a\}, l, inc \uplus \{(a, V_a)\})$, where $V_a \subseteq V$ with $v \in V_a$ iff $a \in fn\,(l(v))$. We define the *application of a substitution* $\{a/x\}$ to $\mathcal{G}$ by $\mathcal{G}\{a/x\} := (V, E, l', inc)$, where $l'(v) := l(v)\{a/x\}$ for all $v \in V$.

The *graph-theoretic interpretation* (1) creates a vertex for every sequential process, (2) takes the restricted names not under prefixes as the edges, and (3) inserts an arc where a name is free in a process. Technically, it is the function $\mathcal{G}[\![-]\!]$ defined by $\mathcal{G}[\![\mathbf{0}]\!] := (\emptyset, \emptyset, \emptyset, \emptyset)$, $\mathcal{G}[\![K\lfloor\tilde{a}\rfloor]\!] := (\{v\}, \emptyset, \{(v, K\lfloor\tilde{a}\rfloor)\}, \emptyset)$, $\mathcal{G}[\![\Sigma_{i\in I\neq\emptyset}\pi_i.P_i]\!] := (\{v\}, \emptyset, \{(v, \Sigma_{i\in I\neq\emptyset}\pi_i.P_i)\}, \emptyset)$, $\mathcal{G}[\![P \mid Q]\!] := \mathcal{G}[\![P]\!] \uplus \mathcal{G}[\![Q]\!]$, and $\mathcal{G}[\![\nu a.P]\!] := \mathcal{G}[\![P]\!] \otimes a$ if $a \in fn\,(P)$, $\mathcal{G}[\![P]\!]$ otherwise.

Structurally congruent processes $P_1 \equiv P_2$ are mapped to equivalent hypergraphs $\mathcal{G}[\![P_1]\!] \approx \mathcal{G}[\![P_2]\!]$. The relation $\approx$ is the smallest equivalence on hypergraphs where replacement of vertex labels by structurally congruent processes is allowed, $(V \uplus \{v\}, E, l \uplus \{(v, P)\}, inc) \approx (V \uplus \{v\}, E, l \uplus \{(v, Q)\}, inc)$, if $P \equiv Q$, and renaming of edges together with the attached processes is possible, $\mathcal{G} \otimes a \approx (\mathcal{G}\{b/a\}) \otimes b$, if $b \notin fn\,(l(v))$ for all $v \in V$. The equivalence $\approx$ preserves the length of the longest simple path, $\mathcal{G}_1 \approx \mathcal{G}_2$ implies $lsp(\mathcal{G}_1) = lsp(\mathcal{G}_2)$.

3.2 Anchored Fragments

By definition, all fragments under a restriction νa share the name a. In *anchored fragments*, we demand that distinguished processes inside the fragments, the *anchors*, share the name a. The corresponding function *anc* gives for a fragment $F_{\mathcal{A}i}$ in $\nu a.(F_{\mathcal{A}1} \mid \ldots \mid F_{\mathcal{A}n})$ the process $anc(F_{\mathcal{A}i}) = P \in \mathcal{S}(F_{\mathcal{A}i})$ that knows the name, i.e., $a \in fn(P)$. When descending an anchored fragment $F_{\mathcal{A}} = \nu a.(F_{\mathcal{A}1} \mid \ldots \mid F_{\mathcal{A}n})$ using the function $nest_\nu$, this guarantees that the vertices labelled by the anchors $anc(F_{\mathcal{A}i})$ are connected via a in $\mathcal{G}[\![F_{\mathcal{A}}]\!]$.

Definition 2. The *set of anchored fragments* $(F_{\mathcal{A}}, G_{\mathcal{A}} \in)\ \mathcal{P}_{\mathcal{A}}$ is defined by

$$F_{\mathcal{A}} ::= K\lfloor\tilde{a}\rfloor \mid \Sigma_{i\in I\neq\emptyset}\pi_i.P_i \mid \nu a.(F_{\mathcal{A}1} \mid \ldots \mid F_{\mathcal{A}n}),$$

where $a \in fn(anc(F_{\mathcal{A}i}))$ for all i, with $anc(K\lfloor\tilde{a}\rfloor) := K\lfloor\tilde{a}\rfloor$, $anc(\Sigma_{i\in I\neq\emptyset}\pi_i.P_i) := \Sigma_{i\in I\neq\emptyset}\pi_i.P_i$, and $anc(\nu a.(F_{\mathcal{A}1} \mid \ldots \mid F_{\mathcal{A}n})) := anc(F_{\mathcal{A}1})$. ◊

Of course, anchored fragments are fragments. We now show that every fragment can be rewritten as an anchored fragment using structural congruence. In the proof, it is important that every sequential process inside a fragment can be chosen as the anchor.

Lemma 2. *Consider $F \in \mathcal{P}_{\mathcal{F}}$ and a process $P \in \mathcal{S}(F)$. Then there is an anchored fragment $F_{\mathcal{A}} \in \mathcal{P}_{\mathcal{A}}$ such that $F_{\mathcal{A}} \equiv F$, $\mathcal{S}(F_{\mathcal{A}}) = \mathcal{S}(F)$, and $anc(F_{\mathcal{A}}) = P$.*

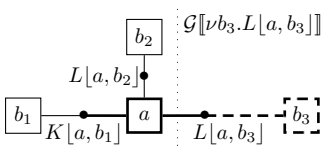
We explain the induction step in the proof of Lemma 2. Given fragment F we compute the standard form $\nu\tilde{a}.(P_1 \mid \ldots \mid P_n)$. Since this does not change the sequential processes, one process P_i is the given process P, wlog. P_1. We split the set of names $\tilde{a}$ into three subsets $\tilde{a}_1, \tilde{a}_2, \tilde{a}_3$ as follows. A name a that is shared by P and $P_2 \mid \ldots \mid P_n$, i.e., $a \in fn(P) \cap fn(P_2 \mid \ldots \mid P_n)$, is in the set $\tilde{a}_1$. A name which is only in the free names of P is in $\tilde{a}_2$. The remaining names are in $\tilde{a}_3$. Shrinking the scopes yields $\nu\tilde{a}_1.(\nu\tilde{a}_2.P \mid \nu\tilde{a}_3.(P_2 \mid \ldots \mid P_n))$. To transform $\nu\tilde{a}_3.(P_2 \mid \ldots \mid P_n)$ into a parallel composition of anchored fragments, we compute the restricted form. It consists of several fragments, $(\nu\tilde{a}_3.(P_2 \mid \ldots \mid P_n))_\nu = G_1 \mid \ldots \mid G_m$. By construction, every G_i contains a process $P_{\mathcal{A}i}$ sharing a name with P. Since each G_i contains less processes than F we can apply the induction hypothesis. This yields anchored fragments $G_{\mathcal{A}i}$ where $anc(G_{\mathcal{A}i}) = P_{\mathcal{A}i}$ shares a name with P. We now have $\nu\tilde{a}_1.(\nu\tilde{a}_2.P \mid G_{\mathcal{A}1} \mid \ldots \mid G_{\mathcal{A}m})$. As the names in $\tilde{a}_1$ are shared by different $G_{\mathcal{A}i}$, we minimise their scopes to get the required anchored fragment.

Example 1. Let $F = \nu b_1, b_2, b_3, a.(K\lfloor a, b_1\rfloor \mid L\lfloor a, b_2\rfloor \mid L\lfloor a, b_3\rfloor)$. We construct the anchored fragment $F_{\mathcal{A}}$ that has $K\lfloor a, b_1\rfloor$ as the anchor, $anc(F_{\mathcal{A}}) = K\lfloor a, b_1\rfloor$. The fragment F already is in standard form. We split the set of names $\{a, b_1, b_2, b_3\}$ into $\tilde{a}_1 = \{a\}$, $\tilde{a}_2 = \{b_1\}$, and $\tilde{a}_3 = \{b_2, b_3\}$. We shrink the scopes of all $\tilde{a}_i$ which gives $\nu a.(\nu b_1.K\lfloor a, b_1\rfloor \mid \nu b_2, b_3.(L\lfloor a, b_2\rfloor \mid L\lfloor a, b_3\rfloor))$. The restricted form of $\nu b_2, b_3.(L\lfloor a, b_2\rfloor \mid L\lfloor a, b_3\rfloor)$ is $\nu b_2.L\lfloor a, b_2\rfloor \mid \nu b_3.L\lfloor a, b_3\rfloor$. Both

fragments, $\nu b_2.L\lfloor a, b_2\rfloor$ and $\nu b_3.L\lfloor a, b_3\rfloor$, are also anchored fragments where the anchors share the name a with $K\lfloor a, b_1\rfloor$. The scope of a is minimal. Our computation returns $\nu a.(\nu b_1.K\lfloor a, b_1\rfloor \mid \nu b_2.L\lfloor a, b_2\rfloor \mid \nu b_3.L\lfloor a, b_3\rfloor)$. ◊

For anchored fragments $F_{\mathcal{A}}$, the nesting of restrictions corresponds to the length of a simple path p in the graph $\mathcal{G}[\![F_{\mathcal{A}}]\!]$, $nest_\nu(F_{\mathcal{A}}) = length(p)$ for some simple path $p \in Paths(\mathcal{G}[\![F_{\mathcal{A}}]\!])$. In the proof, we need that the first element of p is labelled by the anchor of $F_{\mathcal{A}}$, $l(fe(p)) = anc(F_{\mathcal{A}})$. We illustrate the construction of a suitable path p in the induction step. The idea is to extend a path p' that exists by the hypothesis by an edge and a vertex.

Example 2. Consider $F_{\mathcal{A}} = \nu a.(\nu b_1.K\lfloor a, b_1\rfloor \mid \nu b_2.L\lfloor a, b_2\rfloor \mid \nu b_3.L\lfloor a, b_3\rfloor)$.



The figure to the left shows a simple path p in $\mathcal{G}[\![F_{\mathcal{A}}]\!]$ with $length(p) = 2 = nest_\nu(F_{\mathcal{A}})$ and $l(fe(p)) = K\lfloor a, b_1\rfloor = anc(F_{\mathcal{A}})$. By the hypothesis, there is a simple path p' in $\mathcal{G}[\![\nu b_3.L\lfloor a, b_3\rfloor]\!]$ with $length(p') = 1 = nest_\nu(\nu b_3.L\lfloor a, b_3\rfloor)$ and $l(fe(p')) = L\lfloor a, b_3\rfloor = anc(\nu b_3.L\lfloor a, b_3\rfloor)$. This path is $p' = (L\lfloor a, b_3\rfloor, b_3, L\lfloor a, b_3\rfloor)$, depicted by dashed lines. As $\mathcal{G}[\![\nu b_3.L\lfloor a, b_3\rfloor]\!]$ is embedded in $\mathcal{G}[\![F_{\mathcal{A}}]\!]$ (dotted line), p' is a path in $\mathcal{G}[\![F_{\mathcal{A}}]\!]$. The anchor $L\lfloor a, b_3\rfloor$ and the anchor of $F_{\mathcal{A}}$, $K\lfloor a, b_1\rfloor$, are connected with a. We define $p = (anc(F_{\mathcal{A}}), a, p') = (K\lfloor a, b_1\rfloor, a, L\lfloor a, b_3\rfloor, b_3, L\lfloor a, b_3\rfloor)$. It extends p' by the bold lines. ◊

3.3 The Characterisation of Boundedness in Depth

Fragment F is structurally congruent with an anchored fragment $F_{\mathcal{A}}$ (Lemma 2). As $depth(F) \leq nest_\nu(F_{\mathcal{A}}) = length(p)$ for some simple path p in $\mathcal{G}[\![F_{\mathcal{A}}]\!]$ and as the length of the simple paths is bounded, the depth is bounded as well.

Lemma 3. *If $P \in \mathcal{P}$ is bounded in the simple paths by k_{Sim} then P is bounded in depth by k_{Sim} as well.*

It is easy to check that the length of the longest simple path in $\mathcal{G}[\![F]\!]$ is bounded by the nesting of restrictions in F as follows: $lsp(\mathcal{G}[\![F]\!]) \leq 2^{nest_\nu(F)} - 1$. Let F be bounded in depth. There is a fragment $F_D \equiv F$ where the nesting of restrictions is minimal, $nest_\nu(F_D) = min\{nest_\nu(F') \mid F' \equiv F\} = depth(F)$. Since the graphs of F and F_D are equivalent, $lsp(\mathcal{G}[\![F]\!]) = lsp(\mathcal{G}[\![F_D]\!])$ holds. The mentioned inequality and the choice of F_D yield the following lemma.

Lemma 4. *If $P \in \mathcal{P}$ is bounded in depth by k_D then P is bounded in the simple paths by $2^{k_D} - 1$.*

Combined, Lemma 3 and Lemma 4 prove our first main theorem.

Theorem 1. *A process $P \in \mathcal{P}$ is bounded in depth if and only if it is bounded in the simple paths.*

In the following section we understand anchored fragments as trees of bounded height. The boundedness is justified by the following corollary of Lemma 4.

Corollary 1. *Let P be bounded in depth by k_D and $F_{\mathcal{A}} \in Frag\,(Reach\,(P))$ then $nest_{\nu}\,(F_{\mathcal{A}}) \leq 2^{k_D} - 1$.*

4 The Transition Systems of $\mathcal{P}_{BD}$ are Well-structured

Our second main result states that processes of bounded depth have *well-structured transition systems (WSTS)* [10, 1, 11]. A WSTS is a tuple $(S, \rightsquigarrow, \preceq)$, where $(S, \rightsquigarrow)$ is an image finite *transition system* and $\preceq \subseteq S \times S$ is a wqo on the *states* $(s, t \in)\ S$ which is required to be a *simulation*. By definition, the relation $s \preceq t$ is a simulation if state t imitates the transition behaviour of s, i.e., $s \preceq t$ and $s \rightsquigarrow s'$ implies there is t' with $t \rightsquigarrow t'$ and $s' \preceq t'$. To instantiate the framework, we define a qo $\preceq_{\mathcal{P}_{BD}}$ on processes and prove it (1) to be a wqo on $Reach\,(P)$ where P is bounded in depth (Section 4.1) and (2) to be a simulation (Section 4.2). We conclude with a decision procedure for termination.

4.1 A Well-Quasi-Ordering for $\mathcal{P}_{BD}$

Our wqo $\preceq_{\mathcal{P}_{BD}}$ on processes is derived from a wqo on fragments. The idea of the *fragment ordering* $\preceq_{\mathcal{F}}$ is to use the rooted tree embedding and close it under structural congruence. The leafs in Rule (Elem) correspond to elementary fragments: $F_e \preceq_{\mathcal{F}} F_e$ (Rule (1)). Fragment $\nu a.(\Pi_{i\in I} F_i)$ is dominated by $\nu a.(\Pi_{i\in I} G_i \mid \Pi_{j\in J} G_j)$ if the G_i dominate the F_i. This mimics Rule (Comp). If F' is smaller than G' then every $F \equiv F'$ is smaller than $G \equiv G'$ (Rule (3)).

Definition 3. The *fragment ordering* $\preceq_{\mathcal{F}} \subseteq \mathcal{P}_{\mathcal{F}} \times \mathcal{P}_{\mathcal{F}}$ is defined by:

$$(1)\ \frac{}{F_e \preceq_{\mathcal{F}} F_e} \qquad (2)\ \frac{F_i \preceq_{\mathcal{F}} G_i \text{ for all } i \in I}{\nu a.(\Pi_{i\in I} F_i) \preceq_{\mathcal{F}} \nu a.(\Pi_{i\in I} G_i \mid \Pi_{j\in J} G_j)}$$

$$(3)\ \frac{F \equiv F' \preceq_{\mathcal{F}} G' \equiv G}{F \preceq_{\mathcal{F}} G}. \qquad \Diamond$$

Reflexivity of $\preceq_{\mathcal{F}}$ is immediate, transitivity follows from Lemma 8. To relate the fragment ordering $\preceq_{\mathcal{F}}$ with the rooted tree embedding $\preceq_{\mathcal{T}(A)}$, we interpret fragments F as (syntax) trees $\mathcal{T}[\![F]\!]$ as follows: an elementary fragment is a single leaf, $\mathcal{T}[\![F_e]\!] := F_e$, a fragment $\nu a.(F_1 \mid \ldots \mid F_n)$ is the tree $\mathcal{T}[\![\nu a.(F_1 \mid \ldots \mid F_n)]\!] := (a, (\mathcal{T}[\![F_1]\!], \ldots, \mathcal{T}[\![F_n]\!]))$. If we assume that the set A contains the sequential processes and the restricted names in F that are not under prefixes, i.e., $\mathcal{S}(F) \cup rn\,(F) \subseteq A$, then $\mathcal{T}[\![F]\!]$ is a tree over A, $\mathcal{T}[\![F]\!] \in \mathcal{T}(A)$.

If we furthermore assume that A is ordered by the identity, i.e., we consider the qo (A, id), then the rooted tree embedding implies the fragment ordering.

Lemma 5. *Consider the qo (A, id). If $\mathcal{T}[\![F]\!] \preceq_{\mathcal{T}(A)} \mathcal{T}[\![G]\!]$ then $F \preceq_{\mathcal{F}} G$ for all fragments $F, G \in \mathcal{P}_{\mathcal{F}}$.*

To conclude $\preceq_{\mathcal{F}}$ is a wqo from the fact that $\preceq_{\mathcal{T}(A)}$ is a wqo with Lemma 5, (A, id) needs to be a wqo (cf. Lemma 1). This is the case if A is finite. Thus, we need fragments that consist of a finite set of sequential processes and a finite set of restricted names. The idea is to reuse restricted names in parallel compositions, i.e., here we relax the requirement that a name is bound at most once. For every $i \in \mathbb{N}$ we define $con_i : \mathcal{P}_{\mathcal{F}} \to \mathcal{P}_{\mathcal{F}}$ by $con_i(F_e) := F_e$ and $con_i(\nu a.(F_1 \mid \ldots \mid F_n)) := \nu u_i.(con_{i+1}(F_1)\{u_i/a\} \mid \ldots \mid con_{i+1}(F_n)\{u_i/a\})$, where wlog. u_i is fresh for $F_1, \ldots, F_n$. Of course, $F \equiv con_i(F)$ and the restricted names are determined by $nest_\nu(F)$ since $rn(con_i(F)) \subseteq \{u_i, \ldots, u_{i+nest_\nu(F)}\}$.

Example 3. Consider $F_{\mathcal{A}} = \nu a.(\nu b_1.K\lfloor a, b_1\rfloor \mid \nu b_2.L\lfloor a, b_2\rfloor \mid \nu b_3.L\lfloor a, b_3\rfloor)$. We compute $con_0(F_{\mathcal{A}}) = \nu u_0.(\nu u_1.K\lfloor u_0, u_1\rfloor \mid \nu u_1.L\lfloor u_0, u_1\rfloor \mid \nu u_1.L\lfloor u_0, u_1\rfloor)$. $\Diamond$

Following the argumentation above, we now build particular anchored fragments $F_{\mathcal{A}}$ that consist of derivatives where the restricted names are changed by con_0.

Lemma 6. *Let $F \in Frag(Reach(P))$ for some $P \in \mathcal{P}$. There is an anchored fragment $F_{\mathcal{A}} \equiv F$ with $rn(F_{\mathcal{A}}) \subseteq \{u_0, \ldots, u_{nest_\nu(F_{\mathcal{A}})}\}$ and $\mathcal{S}(F_{\mathcal{A}}) \subseteq \{Q\sigma \mid Q \in derivatives(P)$ and $\sigma : fn(Q) \to fn(P) \cup \{u_0, \ldots, u_{nest_\nu(F_{\mathcal{A}})}\}\}$.*

Proof. Let $F \in Frag(Reach(P))$. We recalled that $F \equiv \nu\tilde{a}.(Q_1\sigma_1 \mid \ldots \mid Q_n\sigma_n)$ where $Q_i \in derivatives(P)$ and $\sigma_i : fn(Q_i) \to \tilde{a} \cup fn(P)$ in Section 2. We compute the restricted form, $(\nu\tilde{a}.(Q_1\sigma_1 \mid \ldots \mid Q_n\sigma_n))_\nu =: F'$. It is a fragment F' according to $\widehat{\equiv}$. For F' we compute ${F_{\mathcal{A}}}'$ with Lemma 2. We now have $F \equiv {F_{\mathcal{A}}}'$ and $\mathcal{S}({F_{\mathcal{A}}}') \subseteq \{Q\sigma \mid Q \in derivatives(P)$ and $\sigma : fn(Q) \to \tilde{a} \cup fn(P)\}$.

With the function con_0 we change the restricted names: $con_0({F_{\mathcal{A}}}') \equiv {F_{\mathcal{A}}}'$ and $rn\left(con_0({F_{\mathcal{A}}}')\right) \subseteq \{u_0, \ldots, u_{nest_\nu({F_{\mathcal{A}}}')}\}$. The renaming changes the set of sequential processes. They are now derivatives where the substitutions map into $fn(P) \cup \{u_0, \ldots, u_{nest_\nu({F_{\mathcal{A}}}')}\}$. Thus, $con_0({F_{\mathcal{A}}}')$ satisfies the requirements. $\square$

To see that $\preceq_{\mathcal{F}}$ is a wqo on the reachable fragments of $P \in \mathcal{P}_{BD}$, let k_D be a bound on the depth. We define the set $A := \{u_0, \ldots, u_{2^{k_D}-1}\} \cup \{Q\sigma \mid Q \in derivatives(P)$ and $\sigma : fn(Q) \to fn(P) \cup \{u_0, \ldots, u_{2^{k_D}-1}\}\}$. Obviously, A is finite and thus (A, id) is a wqo.

Let $(F_i)_{i\in\mathbb{N}}$ be a sequence in $Frag(Reach(P))$. Every F_i is structurally congruent with an anchored fragment $F_{\mathcal{A}i}$ in Lemma 6. Corollary 1 yields $nest_\nu(F_{\mathcal{A}i}) \leq 2^{k_D} - 1$. Thus $\mathcal{T}[\![F_{\mathcal{A}i}]\!] \in \mathcal{T}(A)$ with the set A we just defined. The height of $\mathcal{T}[\![F_{\mathcal{A}i}]\!]$ is equal to the nesting of restrictions in $F_{\mathcal{A}i}$. Thus, we have a sequence $(\mathcal{T}[\![F_{\mathcal{A}i}]\!])_{i\in\mathbb{N}}$ of trees in $\mathcal{T}(A)_{2^{k_D}-1}$. According to Lemma 1, $(\mathcal{T}(A)_{2^{k_D}-1}, \preceq_{\mathcal{T}(A)})$ is a wqo and so there are $i < j$ with $\mathcal{T}[\![F_{\mathcal{A}i}]\!] \preceq_{\mathcal{T}(A)} \mathcal{T}[\![F_{\mathcal{A}j}]\!]$. Since A is ordered by the identity, $F_{\mathcal{A}i} \preceq_{\mathcal{F}} F_{\mathcal{A}j}$ with Lemma 5. With Rule (3) we conclude $F_i \preceq_{\mathcal{F}} F_j$. The following lemma holds.

Lemma 7. *Let $P \in \mathcal{P}_{BD}$. Then $(Frag(Reach(P)), \preceq_{\mathcal{F}})$ is a wqo.*

We define the qo $\preceq_{\mathcal{P}_{BD}}$ on $Reach(P)/_{\equiv}$ by $[\Pi_{i\in I} F_i] \preceq_{\mathcal{P}_{BD}} [\Pi_{i\in I} G_i \mid \Pi_{j\in J} G_j]$ if $F_i \preceq_{\mathcal{F}} G_i$ for all $i \in I$. Wqo follows from Lemma 7 and Higman's result.

Proposition 1. *Let $P \in \mathcal{P}_{BD}$. Then $(Reach(P)/_{\equiv}, \preceq_{\mathcal{P}_{BD}})$ is a wqo.*

4.2 The Relation $\preceq_{\mathcal{P}_{BD}}$ is a Simulation

In the proof that $\preceq_{\mathcal{P}_{BD}}$ is a simulation, the following Lemma 8 is crucial. It relates the fragment ordering $F \preceq_{\mathcal{F}} G$ with the standard form of F. This standard form is covered by G in a way that reveals $\preceq_{\mathcal{F}}$ is a simulation.

Lemma 8. *For all $F, G \in \mathcal{P}_{\mathcal{F}}$: $F \preceq_{\mathcal{F}} G$ if and only if $F \equiv \nu\tilde{a}.(P_1 \mid \ldots \mid P_n)$ in standard form and $G \equiv \nu\tilde{a}.(P_1 \mid \ldots \mid P_n \mid R)$ for some $R \in \mathcal{P}$.*

Let $[P] = [\Pi_{i\in I} F_i] \preceq_{\mathcal{P}_{BD}} [\Pi_{i\in I} G_i \mid \Pi_{j\in J} G_j] = [Q]$, which means $F_i \preceq_{\mathcal{F}} G_i$ for all $i \in I$. With Lemma 8 we get $\Pi_{i\in I} F_i \equiv \Pi_{i\in I} \nu a_i.(P_{1_i} \mid \ldots \mid P_{n_i})$. We extrude the names νa_i and check that $[P] \to [P']$ implies $[Q] \to [Q']$ with a case distinction. The direction from right to left in Lemma 8 yields $[P'] \preceq_{\mathcal{P}_{BD}} [Q']$.

Proposition 2. *The relation $\preceq_{\mathcal{P}_{BD}}$ is a simulation on $\mathcal{P}/_{\equiv}$.*

With Proposition 1, Proposition 2, and the fact that $\to$ is image finite up to $\equiv$, we conclude that processes of bounded depth have WSTS.

Theorem 2. *Let $P \in \mathcal{P}_{BD}$. Then $(Reach(P)/_{\equiv}, \to, \preceq_{\mathcal{P}_{BD}})$ is a WSTS.*

4.3 Decidability of Termination for $\mathcal{P}_{BD}$

The WSTS $(S, \rightsquigarrow, \preceq)$ *has a non-terminating computation from $s_0 \in S$* iff an infinite sequence $s_0 \rightsquigarrow s_1 \rightsquigarrow \ldots$ exists. If $\rightsquigarrow$ is effectively computable and $\preceq$ is decidable the following algorithm decides the termination problem [10, 11].

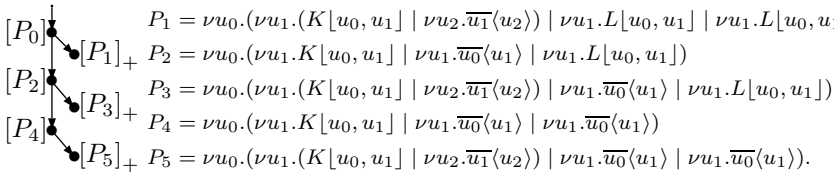
Let $s_0 \in S$. We construct the *finite reachability tree* $FRT(s_0)$. The root is labelled by s_0. For every node labelled by s in the tree, we create a new node for every successor t of s. We connect the node labelled by s and the new node. If there is a node labelled by s' on the path from the root to the new node with $s' \preceq t$, we label the new node by t_+. Otherwise we label it by t. We do not create successors for nodes t_+. The idea is that t with $s' \preceq t$ can simulate the behaviour of s' and thus repeat $s' \rightsquigarrow \ldots \rightsquigarrow t$.

Proposition 3 ([10, 11]). *A WSTS $(S, \rightsquigarrow, \preceq)$ has a non-terminating computation from $s_0 \in S$ if and only if $FRT(s_0)$ contains a node t_+. As $\preceq$ is a wqo, the tree $FRT(s_0)$ is finite and containment of t_+ is decidable.*

The reaction relation is effectively computable and $\preceq_{\mathcal{P}_{BD}}$ is decidable.

Corollary 2. *For $P \in \mathcal{P}_{BD}$ it is decidable whether there is a non-terminating computation starting from $[P]$.*

Example 4. Let $P_0 = \nu u_0.(\nu u_1.K\lfloor u_0, u_1\rfloor \mid \nu u_1.L\lfloor u_0, u_1\rfloor \mid \nu u_1.L\lfloor u_0, u_1\rfloor)$ with $K(x,y) := K\lfloor x, y\rfloor \mid \nu z.\overline{y}\langle z\rangle$ and $L(x,y) := \overline{x}\langle y\rangle$. Then $FRT([P_0])$ is



The root of $FRT([P_0])$ is labelled by $[P_0]$. We have $[P_0] \to [P_1]$ and $[P_0] \to [P_2]$. Thus, we insert two new nodes. For the first node, $[P_0] \preceq_{\mathcal{P}_{BD}} [P_1]$ holds, so we label it by $[P_1]_+$. Since $[P_0] \preceq_{\mathcal{P}_{BD}} [P_2]$ does not hold, the second node is labelled by $[P_2]$. With $[P_2] \to [P_3]$ we construct a new node. As $[P_0] \not\preceq_{\mathcal{P}_{BD}} [P_3]$ but $[P_2] \preceq_{\mathcal{P}_{BD}} [P_3]$, we label it by $[P_3]_+$. The remaining nodes are constructed similarly with $[P_4] \preceq_{\mathcal{P}_{BD}} [P_5]$. The existence of $[P_1]_+$ implies the system has a non-terminating computation from $[P_0]$. ◊

5 Related Work and Conclusion

The interpretation of processes as graphs was proposed in [15, 16] and has been recalled in [17, 18] for the π-Calculus. We related the depth of a process P with a function on the graph $\mathcal{G}[\![P]\!]$. We are not aware of similar results in the literature. The proof required an intricate normal form called anchored fragments.

In [6, 8] decidability of structural congruence relations was investigated. The authors proposed normal forms related with the restricted form in [14]. The standard form of processes is due to [17]. Anchored fragments are more stringent than the normal forms above, and thus reveal more information about the connection structure of process terms.

Finkel generalised the coverability graph procedure for Petri nets to what he called WSTS [10]. He presented algorithms to decide termination and boundedness problems in the general setting. Abdulla et. al. generalised decidability results of temporal properties and simulation relations for lossy channel systems to their notion of WSTS [1]. Both definitions were unified in [11]. This paper is the first to instantiate the WSTS framework for the π-Calculus. Compatibility with the reaction relation required a non-trivial ordering $\preceq_{\mathcal{P}_{BD}}$.

Based on a translation of π-Calculus into multisets, orderings on processes defined by multiset containment relations were studied in [7]. We considered the more intricate wqos, i.e., $\preceq_{\mathcal{P}_{BD}}$ needed to be well-behaved under reaction.

In [19, 5] type systems for the π-Calculus were presented that ensure termination of well-typed processes. We observe that terminating processes are always bounded in depth due to the finite number of reachable processes. Furthermore, our result is more general in that we instantiate the WSTS framework for $\mathcal{P}_{BD}$ and then derive decidability of termination as a corollary. To turn our decidability result into a practical procedure, approximations on $\preceq_{\mathcal{P}_{BD}}$ should be developed to prune the finite reachability tree.

References

1. P. A. Abdulla, K. Čerans, B. Jonsson, and Y.-K. Tsay. Algorithmic analysis of programs with well quasi-ordered domains. *Information and Computation*, 160(1–2):109–127, 2000.
2. R. M. Amadio and C. Meyssonnier. On decidability of the control reachability problem in the asynchronous π-calculus. *Nordic Journal of Computing*, 9(1):70–101, 2002.
3. L. Caires. Behavioural and spatial observations in a logic for the π-Calculus. In *FOSSACS 2004*, volume 2987 of *LNCS*, pages 72–89. Springer-Verlag, 2004.
4. M. Dam. Model checking mobile processes. *Information and Computation*, 129(1):35–51, 1996.
5. Y. Deng and D. Sangiorgi. Ensuring termination by typability. *Information and Computation*, 204(7):1045–1082, 2006.
6. J. Engelfriet and T. Gelsema. Multisets and structural congruence of the pi-calculus with replication. *Theoretical Computer Science*, 211(1–2):311–337, 1999.
7. J. Engelfriet and T. Gelsema. Structural inclusion in the pi-calculus with replication. *Theoretical Computer Science*, 258(1–2):131–168, 2001.
8. J. Engelfriet and T. Gelsema. A new natural structural congruence in the pi-calculus with replication. *Acta Informatica*, 40(6):385–430, 2004.
9. G.-L. Ferrari, S. Gnesi, U. Montanari, and M. Pistore. A model-checking verification environment for mobile processes. *ACM Transactions on Software Engineering and Methodology*, 12(4):440–473, 2003.
10. A. Finkel. Reduction and covering of infinite reachability trees. *Information and Computation*, 89(2):144–179, 1990.
11. A. Finkel and Ph. Schnoebelen. Well-structured transition systems everywhere! *Theoretical Computer Science*, 256(1–2):63–92, 2001.
12. A. Habel. *Hyperedge Replacement: Grammars and Languages*, volume 643 of *LNCS*. Springer-Verlag, 1992.
13. G. Higman. Ordering by divisibility in abstract algebras. *Proc. London Math. Soc.* (3), 2(7):326–336, 1952.
14. R. Meyer. A theory of structural stationarity in the π-Calculus. *Under revision*, 2008.
15. G. Milne and R. Milner. Concurrent processes and their syntax. *JACM*, 26(2):302–321, 1979.
16. R. Milner. Flowgraphs and flow algebras. *JACM*, 26(4):794–818, 1979.
17. R. Milner. *Communicating and Mobile Systems: the π-Calculus*. Cambridge University Press, 1999.
18. D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.
19. N. Yoshida, M. Berger, and K. Honda. Strong normalisation in the π-Calculus. *Information and Computation*, 191(2):145–202, 2004.