Karlsruhe Institute of Technology

Solving Power Domination with Bounded Propagation

Bachelor's Thesis of

Christoph Niederbudde

At the Department of Informatics
Institute of Theoretical Informatics (ITI)

Reviewer:	T.T.-Prof. Dr. Thomas Bläsius
Second reviewer:	PD Dr. Torsten Ueckerdt
Advisor:	Max Göttlicher

28.11.2024 – 21.03.2025

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

I declare that I have developed and written the enclosed thesis completely by myself. I have not used any other than the aids that I have mentioned. I have marked all parts of the thesis that I have included from referenced literature, either in their original wording or paraphrasing their contents. I have followed the by-laws to implement scientific integrity at KIT.

. . . . . .

Abstract

We analyze the problem l-round Power Dominating Set as an intermediate problem between Dominating Set and Power Dominating Set. It is motivated by placing phasor measurement units in a power grid to observe its state with bounded delay. Our contribution is threefold. We analyze the parametrized complexity of l-round PDS an prove that it is $W[2l]$ complete for arbitrary fixed l. We introduce reduction rules that can simplify a graph without changing the size of the minimal solution. We also apply the implicit hitting set approach to l-round PDS and thus provide a new algorithm to solve this problem. We implement and evaluate this algorithm on known instances.

Zusammenfassung

Wir analysieren l-round Power Dominating Set als Zwischenproblem zwischen Dominating Set und Power Dominating Set. Das Problem ist von der Platzierung von Messgeräten in Stromnetzen motiviert. Das Ziel ist es, eine minimale Menge von Knoten im Netz zu finden, die den vollständigen Zustand des Netzes mit beschränkter Verzögerung messen können. Wir liefern drei Beiträge zum Verständins diese Problems.

Wir analysieren die parametrisierte Komplexität von l-round PDS und zeigen, dass es für beliebige, konstante l $W[2l]$ vollständig ist. Außerdem führen wir Reduktionsregeln ein, die einen Graph vereinfachen können ohne die Größe der idealen Lösung zu verändern. Außerdem reduzieren wir l-round PDS auf Hitting Set. Basierend auf dem impliziten Hitting Set Ansatz stellen wir einen neuen Algorithmus zum Lösen von l-round PDS vor und implementieren und testen diesen auf Standardinstanzen.

Contents

1. Introduction

1.1. Motivation

In order to monitor power voltage and current in electrical grids, grid operators use so-called phasor management units or PMUs that can monitor the power flow in a grid node. Because these PMUs are expensive, it is important to select a good set of grid nodes to install PMUs at. The underlying graph problem of finding a minimal set of vertices that can observe the entire graph is called Power Dominating Set and was first formalized by Baldwin et al.[BMBA93]. The current formulation of the problem was introduced by Brueni [Bru93] and allows observation of vertices by two rules. Firstly, every sensor observes its vertex and all its neighbors. Secondly, if a vertex and all its neighbors except for one are observed, the observation can *propagate* to the last remaining neighbor and observe it as well. While placing PMUs according to a solution for Power Dominating Set allows to measure the state of a static grid, it gives no guarantee for the delay between a change in the grid and the change registering in a sensor. Aazami[Aaz10] introduce l-round Power Dominating Set to mitigate this problem by limiting the distance of propagation, which is equivalent to the number of vertices a signal has to pass before reaching a sensor. Vertices become observed for l-round PDS in up to l parallel rounds. In the first round, all sensors observe their vertices and neighbors. In subsequent rounds, the observation can propagate in parallel. This problem is also interesting as an intermediate problem between Dominating Set and Power Dominating Set as it is identical to Dominating Set for $l = 1$.

1.2. Related Work

Aazami provide several theoretical results for l-round PDS. They show that it is NP-complete even for planar graphs and provide a limit to the approximation of l-round PDS. They also provide different algorithms to solve it. They propose a dynamic programming algorithm for l-round PDS based on tree decomposition. Unfortunately, that algorithm is exponential in both the tree width of the graph and the logaritm of l. They also provide a PTAS for l-round PDS on planar graphs and an integer linear programming formulation for l-round PDS. Brimkov et al. [BMS19] also provide an integer linear programming formulation for Power Dominating Set that allows restrictions for bounded propagation. While they only evaluate its efficiency for finding minimal regular Power Dominating Sets, the formulation can also solve l-round PDS. Additionally, Ciao [Lia16] provide an algorithm that can solve l-round PDS in linear time on trees and block graphs.

There are more practical results for solving regular Power Dominating Set. The current state-of-the-art for PDS is reducing it to the hitting set problem. This is done by finding certain subsets of vertices called *forts* that can not be reached by propagation. A set of vertices is a solution for PDS if and only if it contains a neighbor of every fort of the graph. The number of forts can be exponential, so it is impractical to calculate all forts. Instead, the *implicit hitting set approach* is used. This approach starts with a subset of forts and calculates a minimal

hitting set. If this hitting set is not a solution for PDS, more forts are generated and added to the set of forts. This process is repeated until a solution for PDS is found. The implicit hitting set approach was introduced for PDS by Brimkov et al. [BFH18] and later improved by Smith et al. [SH20] and most recently by Bläsius et al. [BG25]. Bläsius et al. combine this approach with reduction rules that simplify the graph in a pre-processing step and provide the current state-of-the-art for solving PDS.

1.3. Outline

We make three contributions to understanding and solving l-round PDS. In Chapter 3, we study the parametrized complexity of l-round PDS when parametrized by solution size based on l. For the case $l = 1$, l-round PDS is identical to Dominating Set which is known to be $W[2]$ complete [DFR98]. For $l = n$ it is identical to Power Dominating Set which is known to be $W[P]$ complete [BG25]. We show that l-round PDS is $W[2l]$ complete for constant arbitrary l. For this, we reduce from Weighted Monotone t-Normalized Satisfiability, which is known to be $W[t]$ complete for circuits of depth t [DFR98 | KTX17]. We also introduce l-round Extended Power Dominating Set as an intermediate problem to help with the reduction. We provide formulations of these problems together with a basic introduction to parametrized complexity in Chapter 2

In Chapter 4, we provide an algorithm for solving l-round Extended Power Dominating Set. For our second contribution, we provide reduction rules for l-round PDS similar to those provided by Bläsius et al. They simplify a given instance by either removing edges or vertices or limiting the selection of vertices. For that, they introduce an extended version of PDS which allows for vertices to be either pre-selected for or excluded from the solution. We use the reduction rules with our variation l-round EPDS.

Lastly, we apply the implicit hitting set approach to l-round PDS. For this, we introduce a more general version of forts which we call *base sets* because forts alone are not enough to solve l-round PDS. We test different heuristics to find these base sets and evaluate their effectiveness based on our experiments. We also test how the implicit hitting set approach performs based on the value of l.

2. Preliminaries

In this chapter, we give an overview over the concepts used in the remaining chapters. We first provide a brief introduction to parameterized complexity and the W hierarchy. We then give formal definitions of l-round Power Dominating Set, the extension variant l-round Extended Power Dominating Set and the problem Weighted Monotone t-Normalized Satisfiablity, which we use for our complexity analysis in Chapter 3.

2.1. Parametrized complexity and the W hierarchy

We only give a short introduction to parameterized complexity and the W hierarchy. For more details, see [DFR98].

Parametrized complexity allows for a more fine-grained view on the complexity of NP-hard problems. For this, the hardness of a problem is expressed not only based on the size of the instance but also on some other parameter such as the size of the optimal solution. Formally, a parametrized problem is a language $\mathcal{L} \in \Sigma \times \mathbb{N}$. An instance of $\mathcal{L}$ has the form (x, k) with $x \in \Sigma$ and parameter k.

The parametrized complexity of different problems can be compared using a parametrized reduction. This is a reduction that maps an instance (x, k) of one problem to an instance (x', k') of a second problem with $k' \leq f(k)$. The reduction has to be possible in $g(k) \cdot h(|x|)$ for computable functions f, g and a polynomial function h[DFR98]. A problem is called *fixed parameter tractable* (FPT) if it can be solved in $O(f(k) * n^c)$ for a computable function f and fixed c.

The W hierarchy is a hierarchy of the parametrized complexity of different NP hard problems with $FPT \subseteq W[1] \subseteq ... \subseteq W[P]$. The base problem that defines the W hierarchy is Weighted Circuit Satisfiability. A circuit is represented as a directed acyclic graph with several input nodes with input degree 0, negation nodes with input degree 1, conjunction and disjunction nodes with input degree of at least 2 and one output node with output degree 1. Input nodes can be assigned the values 0 or 1. The values of the remaining nodes are calculated based of the inputs and the boolean function of the circuit. An assignment of values is satisfying if the value of the output node is 1. An instance (C, k) consists of a circuit C and parameter k. It asks whether there is an satisfying assignment that sets precisely k input nodes to true.

The weft of a circuit is the maximum number of and-gates or or-gates on any directed path from an input node to the output node that have an input degree larger than 2. The problem WCS$[t]$ is Weighted Circuit Satisfiablility restricted to circuits with weft t. A problem is in $W[t]$ if it can be reduced to WCS$[t]$ with a parametrized reduction. A problem is in $W[P]$ if it can be reduced to WCS with a parametrized reduction. Conversely, a problem is $W[t]$ hard or $W[P]$ hard if it can be reduced to from WCS$[t]$ or from WCS.

2.2. Weighted Monotone t-Normalized Satisfiablity

Weighted Monotone t-Normalized Satisfiablility (WSAT$^+[t]$) is a variant of Weighted Circuit Satifiability restricted to normalized monotone circuits. A monotone circuit consists only of and-nodes and or-nodes. Furthermore, circuits of WSAT$^+[t]$ are restricted by depth rather than by weft. This problem is known to be $W[t]$ complete for even t [DFR98 | KTX17].

WSAT$^+[t]$ An instance of WSAT$^+[t]$ consists of a normalized circuit of depth t and parameter k. The circuit is split in $t+2$ layers L_i. The layer L_0 contains the input nodes. Layers L_i for $1 \leq i \leq t$ alternate between and-layers and or-layers, with and-layers containing only and-nodes and or-layers containing only or-nodes. The layer L_t is always an and-layer. For our proof, we only consider circuits of even depth, so even layers contain and-nodes and odd layers contain or-nodes. The layer L_{t+1} contains only the output node. All and-nodes have an output degree of 1. Furthermore, we only allow edges between layers L_i and L_{i+1}. The parameter k is the size of the solution.

This definition differs slightly from known definitions like [DFR98 | KTX17]. However, the definition is equivalent as additional requirements of our definition can be easily met by further normalization. The requirements added by our definition are that edges cannot skip layers and the circuit cannot have a depth smaller than t. Some definitions also do not restrict the output degree of nodes. If the initial circuit has edges that skip layers, we can simply add intermediate nodes to the skipped layers that have input and output degrees of one. If the circuit has a depth of less than t, we can add intermediate layers that only consist of one node. If the circuit contains an and-node v with an output degree of $r > 1$, we can replace it with r copies that each have the same inputs and are each connected to one of the output nodes of v. This normalization adds at most $O(n^2)$ nodes to the graph and can also be performed in time $O(n^2)$.

2.3. l-round Power Dominating Set

Graphs and Neighborhoods Let $G = (V, E)$ be a graph with undirected edges. We call $N(v) = \{u : \{u, v\} \in E\}$ the open neighborhood of v and $N[v] = N(v) \cup \{v\}$ the closed neighborhood of v.

Power Dominating Set An instance of Power Dominating Set is an undirected graph $G = (V, E)$. The goal of PDS is to find a minimum set $S \subseteq V$ such that all vertices can be observed by propagation. Vertices can be observed by two rules.

- Domination rule: A vertex is observed if it is in the open neighborhood of a vertex in S.
- Propagation rule: Let $u, v \in V$ be vertices of G and let u be in the closed neighborhood of v. If $N[v] \setminus \{u\}$ is observed, then u can become observed. We say that v is propagating its observation status to u.

l-round Power Dominating Set Like PDS, an instance of l-round PDS consists of a graph G and looks for a minimal set S to observe the graph. The observation of vertices works similar to regular PDS but takes place in up to l parallel rounds.

Intuitively, parallel propagation works as follows: In the first round of propagation, the domination rule is applied to all vertices in S. In each subsequent round, the propagation rule is applied to every observed vertex that is able to propagate in parallel.

Aazami [Aaz10] provide a formal definition for a set $\mathcal{P}^i(S)$ of vertices that can be power dominated after i rounds of propagation by selecting a set S. We slightly modify this definition and also add a definition for a set $\mathcal{Q}^i(S)$ of vertices that can propagate to some other vertex after i rounds of propagation.

Definition 2.1: *Given a graph $G = (V, E)$ and a subset $S \subset V$ of selected vertices, the set of vertices that can propagate after at most i rounds of propagation, denoted $\mathcal{Q}^i(S)$, is defined as*

$$\mathcal{Q}^i(S) = \{v \in \mathcal{P}^i(S) : \exists u \in N(v) : N[v] \backslash u \subseteq \mathcal{P}^i(S)\} \tag{2.1}$$

The set of vertices that can be power dominated by applying at most i rounds of parallel propagation is defined recursively as follows:

$$\mathcal{P}^i(S) = \begin{cases} \bigcup_{v \in S} N[v] & \text{if } i = 1 \\ \mathcal{P}^{i-1}(S) \bigcup \{v : \{u, v\} \in E, u \in \mathcal{Q}^{i-1}(S) & \text{if } i \geq 2 \end{cases} \tag{2.2}$$

For $i = 1$, $\mathcal{P}^1$ matches the domination rule, for $i \geq 2$ the set of observed vertices is extended according to the propagation rule. We also use the following formal definition for l-round PDS from [Aaz10].

Definition 2.2: *Given a graph $G = (V, E)$ and a parameter l, a set $S \subseteq V$ is a l-round Power Dominating Set if $\mathcal{P}^l(S) = V$.*

2.4. l-round Extended Power Dominating Set

The goal of l-round EPDS is to allow more restrictions of vertex selection and propagation to an instance. In addition to the graph G, an instance of l-round EPDS also consists of

- A set of propagating vertices $W \subseteq V$. The propagation rule can only be applied to vertices in W.
- A set of pre-selected vertices $X \subseteq V$. Any solution S must be a superset of X.
- A set of selectable vertices $Y \subseteq V$. Only vertices in Y can be selected for S.
- A set of pre-observed vertices $Z \subseteq V$. These vertices are always observed after the first round of propagation independently of the domination rule.
- A set of target vertices $T \subseteq V$. We only require the target vertices to be observed after l rounds of propagation rather than all vertices in V.

The sets of observed vertices and vertices that can propagate after i rounds are defined as follows:

$$\mathcal{Q}^i(S) = \{v \in \mathcal{P}^i(S) : v \in W \wedge \exists u \in N(v) : N[v] \backslash u \subseteq \mathcal{P}^i(S)\} \tag{2.3}$$

$$\mathcal{P}^i(s) = \begin{cases} \bigcup_{v \in S} N[v] \cup Z & \text{if } i = 1 \\ \mathcal{P}^{i-1}(S) \bigcup \{v : \{u, v\} \in E, u \in \mathcal{Q}^{i-1}(S)\} & \text{if } i \geq 2 \end{cases} \tag{2.4}$$

The l-round EPDS problem is formally defined as follows:

Definition 2.3: *l-round EPDS: Given a graph $G = (V, E)$, sets $W, X, Y, Z, T \subseteq V$ and a parameter l, a set $S \subseteq V$ is a l-round Extended Power Dominating Set if $S \subseteq Y$, $X \subseteq S$ and $\mathcal{P}^l(S) \supseteq T$.*

Solution candidates For an instance $(G = (V, E), W, X, Y, Z, T)$, we call any set of vertices $S \subseteq V$ a solution candidate for G. We call S *valid* if $S \subseteq Y$ and $X \subseteq S$.

3. Complexity Analysis

In this chapter we show that l-round Power Dominating Set and l-round Extended Power Dominating Set are $W[2l]$ complete when parametrized by solution size. For this, we provide parametrized reductions from WSAT$^+[2l]$ to l-round EPDS (Section 3.1), from l-round EPDS to l-round PDS (Section 3.2) and from l-round PDS to WSAT$^+[2l]$ (Section 3.3). This fully proves the parametrized complexity of l-round PDS because WSAT$^+[t]$ is know to be $W[t]$ complete for even t [DFR98].

3.1. Reduction from WSAT$^+[2l]$ to l-round EPDS

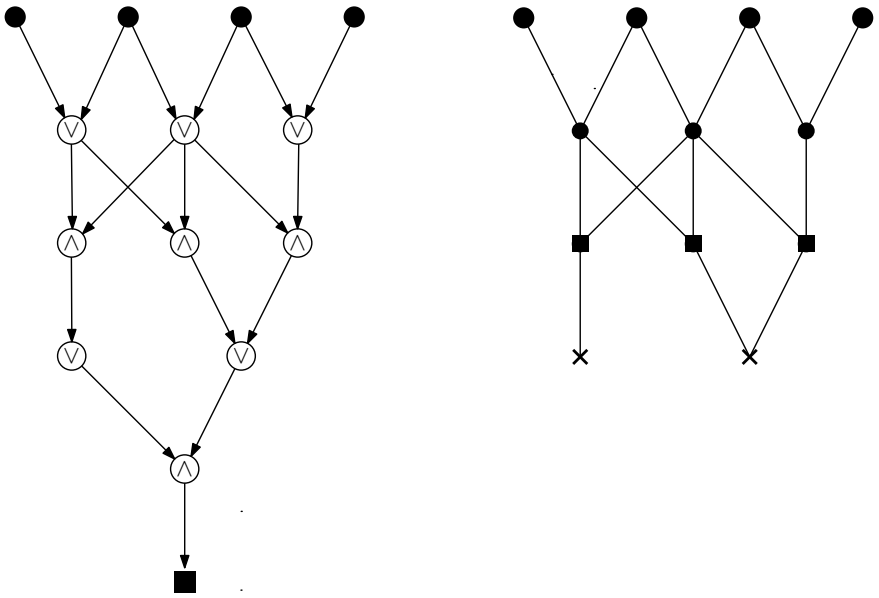
We provide a parametrized reduction from WSAT$^+[2l]$ to l-round EPDS to show that l-round EPDS is $W[2l]$ hard. We construct an instance of l-round EPDS that can simulate a circuit. Intuitively, we simulate or-gates over observation and and-gates over propagation since a vertex can propagate if all of its neighbors except for one are observed and a vertex can become observed if any of its neighbors is selected or can propagate.

Construction 3.1: *For an instance $(G' = (V', E'), k)$ of WSAT$^+[2l]$ with $V' = \bigcup_{i=0,\dots,2l+1} L_i$ we define an instance of l-round EPDS with graph G as follows:*

- *For each input node of $v' \in L_0$ we add a selectable vertex v to G*
- *For each or-node $v' \in L_i$ for $1 \leq i < 2l - 1$ we add a basic vertex v to G. For each or-node $u' \in L_{2l-1}$ we add a target vertex u to G*
- *For each and-node $v' \in L_i$ for $2 \leq i < 2l$ we add a pre-observed and propagating vertex v to V. The and-node in L_{2l} is not added to G.*
- *The output node v_{out} is not added to G*
- *The edges connecting nodes in G' are added to G as undirected edges.*

3.1.1. Proof of Equivalence

Because we add exactly one selectable vertex for every input node, a valid solution candidate for the construction translates to a solution candidate for the circuit and a solution candidate in the circuit translates to a valid solution candidate in the construction. To show that the construction simulates the circuit correctly, we first show that the vertices that simulate or-nodes in layer $2i + 1$ for some i are observed after i rounds for some solution candidate if and only if the respective or-node would output true for the equivalent solution candidate for the circuit. Using this, we can show that all target vertices in the construction are observed if and only if the entire circuit outputs true.



a) Normalized circuit with alternating layers of *and* and *or* gates

b) Constructed graph for l-round PDS. Only the top vertices can be selected. The square vericties are pre-observed and the × vertices are target vertices.

Figure 3.1.: Example transformation from a normalized circuit to a l-round EPDS instance. We replace the and-gates with pre-observed vertices, the or-gates with regular vertices and the input nodes with selectable vertices. The last and-gate and output node are removed.

Definition 3.2: *Let $(G' = (V', E'), k)$ be an instance of WSAT$^+$[2l], $G = (V, E)$ an instance of l-round EPDS and $I : V' \to V$ a (partial) function from nodes of G' to vertices in G. Let X' be a solution candidate for G' and $S = I(X')$ a solution candidate for G. I is called valid for round $1 \leq i \leq l$ if:*

- *I is defined on L_{2i-1} and any node $v' \in L_{2i-1}$ outputs true if and only if $v = I(v') \in \mathcal{P}^i(S)$*
- *I is not defined on L_{2i} or any node $v' \in L_{2i}$ outputs true if and only if $v = I(v') \in \mathcal{Q}^i(S)$*
- *For every $j > 2i$: $I(L_j) \cap \mathcal{P}^i(S) = I(L_j) \cap \mathcal{Q}^i(S) = \emptyset$*

For an instance (G', k) of WSAT$^+$[2l] and the instance G of l-round PDS obtained by Construction 3.1, we can define a partial function I that projects every node of G' that is added to G to the equivalent vertex in G. It is easy to see that for every solution candidate X' of G', $S = I(X')$ is also a solution candidate of G as it consists only of selectable vertices.

Lemma 3.3: *Let $(G' = (V', E'), k)$ be an instance of WSAT$^+$[2l], $(G = (V, E), W, X, Y, Z, T)$ the instance of l-round PDS obtained by Construction 3.1 and I the respective function. Let X' be a solution candidate for G' and $1 \leq i \leq l$. If the first and third requirements of Definition 3.2 are true then I is valid.*

Proof. If $i = l$, I is not defined on L_{2i} as the nodes of the last layer are not added to G in the construction. Assume the first and third requirement to be true for some $1 \leq i < l$. Let u' be a vertex of L_{2i} and $u = I(u')$. By definition, u' must be an and-node. u' has input vertices $v'_1, ..., v'_n \in L_{2i-1}$ and one output vertex $o' \in L_{2i+1}$.

$N[u] = I(\{u', o', v'_1, .., v'_n\}) = \{u, o, v_1, ..., v_n\}$. Because u is a pre-observed vertex, we know that $u \in \mathcal{P}^{2i}(S)$. Since $o' \in L_{2i+1}$ we know from the third requirement that $o \notin \mathcal{P}^{2i}(S)$.

Assume u' outputs true. This means that all input nodes $v'_1, ..., v'_n$ of u' also output true. Using the first requirement, we know that $v_1, ..., v_n \in \mathcal{P}^i(S)$, so $N[u] \setminus o = \{u, v_1, ..., v_n\} \subseteq \mathcal{P}^i(S)$ and $u \in \mathcal{Q}^i(S)$.

Assume u' outputs false. This means that some input node $v'_k \in v'_1, ..., v'_n$ of u' also outputs false. From the first requirement, we know that $v_k \notin \mathcal{P}^i(S)$. Thus, u has at least two neighbors in o and v_k that are not in $\mathcal{P}^i(S)$, so $u \notin Q^i(S)$. ∎

Using Lemma 3.3 we can show that the observation in the constructed graph behaves equivalently to the initial circuit. For this, we prove that the respective function I is always valid.

Lemma 3.4: *Let $(G' = (V', E'), k)$ be an instance of WSAT⁺[2l], $(G = (V, E), W, X, Y, Z, T)$ the instance of l-round PDS obtained by Construction 3.1 and I the respective function. Then I is valid for every $1 \leq i \leq l$.*

Proof. We prove this lemma by induction over i.

Induction start i=1: Let $u' \in L_1$ be a vertex and $u = I(u')$. By definition, u' must be an or-node.

Assume u' outputs true. This means that u' has an input vertex $v' \in L_0$ that outputs true. Because L_0 is the input layer, v' must be in the solution candidate X'. This means that u can be observed by applying the domination rule to $I(v') \in S$, so $u \in \mathcal{P}^1(S)$.

Assume u' outputs false. That means that all input nodes of u' are not in the solution candidate X'. Let v' be a node and $v = I(v')$ be a neighbor of $u = I(u')$ in G. Then v' is either an input node or an output node of u'. If v' is an input node, $v' \notin X$ and $v \notin S$. If v' is an output node of u' then v is not selectable, and $v \notin S$. This means that u is not observed by the domination rule and $u \notin \mathcal{P}^1(S)$.

For any $j > 2$ and $u' \in L_j$, all input nodes must be in layer L_{j-1} and all output nodes must be in layer L_{j+1}. Since $j - 1 > 0$ and $S \subseteq I(S_0)$, we know that $u = I(u')$ does not have a neighbor in S. Thus $u \notin N[v]$ for any $v \in S$, so $u \notin \mathcal{P}^1(S)$.

We can conclude that the first and third requirements hold for $l = 1$,so we know from Lemma 3.3 that I is valid for $l = 1$.

Induction step $i \to i + 1$. Assume I to be valid for some $i < l$. Let $u' \in L_{2i+1}$ be a vertex and $u = I(u')$. By definition, u' must be an or-node.

Assume u' outputs true. This means that u' has an input vertex $v' \in L_{2i}$ that outputs true. Since I is valid for i we can assume that $v = I(v') \in \mathcal{Q}^i(S)$. This means that u is not in $\mathcal{P}^{i+1}(S)$

Assume u' outputs false. That means that all input nodes $v' \in L_{2i}$ of u' output false. Since I is valid for i we can assume that $I(v') \notin \mathcal{Q}^i(S)$ for any input node v' of u'. All output nodes o' of u' are in layer L_{2i+2}, so we know that $I(o') \notin \mathcal{Q}^i(S)$. This means that $N[u] \cap \mathcal{Q}^i(S) = \emptyset$. We also know that $u \notin \mathcal{P}^i(S)$ because $u \in I(L_{2i+1})$, so $u \notin \mathcal{P}^{i+1}(S)$.

Any vertex $u \in I(L_j)$ for $j > 2i + 2$ is connected only to vertices in $I(L_{j-1})$ and $I(L_{j+1})$. Since $j - 1 > 2i$ we know that they cannot propagate in round i, so u is not observed in round $i + 1$.

We can conclude that the first and third requirements for I being valid hold for round $i + 1$ and know from Lemma 3.3 that I is valid for round $i + 1$. ∎

Lemma 3.5: *Let (G', k) be an instance of WSAT$^+$[2l] and (G, W, X, Y, Z, T) be the instance of l-round PDS obtained by Construction 3.1. G' has a solution if and only if G has a solution of size k.*

Proof. Let $X' \subseteq L_0$ be a solution of G' with $|X'| = k$. That means that the and-node in layer $2l$ and the or-nodes in layer $2l-1$ all output true. Since all vertices in $I(L_0)$ are selectable and G does not have any pre-selected vertices, $S = I(X')$ is a valid solution candidate for G. From Lemma 3.4 we know that $I(L_{2l-1}) \subseteq \mathcal{P}^l(S)$. That means that all target vertices are observed after l rounds, so S is a solution of size k for G.

Let $S \subseteq Y$ be a solution of size k of G. Since $I(L_0)$ are the only selectable vertices in G, there is a solution candidate $X' \subseteq L_0$ for G' such that $S = I(X')$. The vertices in $I(L_{l-1})$ are target vertices, so they must be observed after l rounds of propagation. From Lemma 3.4 we know that all or-nodes in layer L_{l-1} output true in G' and $|X'| = |S| = k$, so the and-node in L_l also outputs true and X' is a solution for G'. ∎

Corollary 3.6: *There is a parametrized reduction from WSAT$^+$[t] to l-round EPDS by applying the construction we presented and solving the instance of l-round EPDS. The construction can be done in time $O(n)$ and the solution size does not change.*

3.2. Reduction from l-round EPDS to l-round PDS

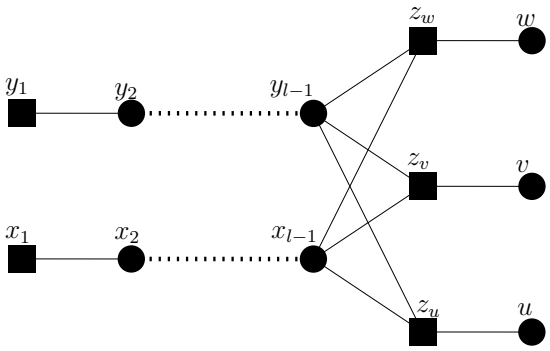
In this section, we show that l-round PDS and l-round EPDS have the same parametrized complexity. It is easy to see that l-round PDS can be reduced to l-round EPDS since every instance $G = (V, E)$ of l-round PDS is also an instance of l-round EPDS with $W = Y = T = V$ and $Z = X = \emptyset$. To provide a parametrized reduction from l-round EPDS to l-round PDS, we transform an arbitrary instance of l-round EPDS to an instance with $W = Y = T = V$ and $Z = X = \emptyset$. We call these instances *normalized*. Each transformation step will be possible in polynomial time, and the transformed instance will have a solution of size $k + c$ if and only if the initial instance has a solution of size k for fixed c.

3.2.1. Normalize propagating vertices

For any instance of l-round EPDS with non-propagating vertices, we provide an equivalent instance with only propagating vertices. We achieve this by adding a leaf x_v to every non-propagating vertex v. The leaf is propagating, not pre-observed, selectable or pre-selected, and not a target vertex. This construction can be performed in $O(n)$, and we will show that it does not change the solutions of the instance. The construction and the following proof are adapted from [BG25].

Lemma 3.7: *Let $(G' = (V', E'), W', X', Y', Z', T')$ be an instance of l-round EPDS with some non-propagating vertices, and let $(G = (V, E), W = V, X', Y', Z', T')$ be the instance obtained by the detailed construction with only propagating vertices. Then G has a solution of size k if and only if G' has a solution of size k for any $k \in \mathbb{N}$.*

Proof. Let $S \subseteq Y'$ be a solution candidate for G' or G. We know that $x_v \notin S$ for any added leaf x_v as the added leafs are not selectable. This means that any selected vertex is present in both G' and G. If a selected vertex v is propagating in G', the domination rule can be applied the same way for G' and G since the neighborhood does not change. If v is not propagating in G', the domination rule also observes the new vertex x_v in G. However, this has no additional effects since x_v is a leaf and cannot propagate to any vertex other than v.



Target gadget for three non-target vertices u, v and w

Figure 3.2.: Target gadget to turn non-target vertices into target vertices. The squares are pre-observed vertices. The paths x_i and y_i are only added once while we add a pre-observed vertex z_u for every non-target vertex u.

Let v be a vertex that is propagating in G'. If the propagation rule can be applied to v in G' or G, it can also be applied in the same way in the other graph since the neighborhood of propagating vertices is not changed by the transformation.

Let v be a vertex that is not propagating in G'. Then the propagation rule cannot be applied to v in G'. The propagation rule could be applied to v in G. However, v can only propagate to the new leaf x_v since a vertex can only propagate if it has exactly one unobserved neighbor. If v is not selected, the only way for x_v to become observed is by propagation from v, so v cannot propagate until all its neighbors other than x_v are observed. Since x_v is a leaf, this added propagation does not have any additional effects.

In conclusion, a vertex that is present in G' is observed after l rounds in G' if and only if is also observed in G after l rounds for any set of selected vertices $S \subseteq Y'$. Since the transformation does not add or remove any target vertex, a set $S \subseteq Y'$ is a solution for G' if and only if it is also a solution for G. ∎

3.2.2. Normalize target vertices

For any instance of l-round EPDS with non-target vertices, we provide a construction to obtain an equivalent instance with only target vertices without adding non-propagating vertices. The construction can be performed in $O(l + n) = O(n)$ for a constant l and does not change the solutions of the instance.

Construction 3.8: *Let $(G' = (V', T'), W', X', Y', Z', T')$ be an instance of l-round EPDS with some non-target vertices $u_1, ..., u_k \notin T'$. We define an instance $(G = (V, E), W, X', Y', Z, T = V)$ that is equivalent to G' and has the same set of selectable vertices but does not have any non-target vertices.*

For $l = 1$ we can make every vertex that is not a target vertex pre-observed, so all vertices except for the target vertices are observed after the first round.

For $l >= 2$ we add two pre-observed vertices x_1 and y_1 that are each connected to a path $x_2, ..., x_{l-1}$ or $y_2, ..., x_{l-1}$ of length $l - 2$. For every non-target vertex u we add a pre-observed vertex z_u that is connected to u, x_{l-1} and y_{l-1}. All added vertices are propagating and not selectable. We then make u a target vertex.

Claim 3.9: *For each $1 \leq i \leq l-1$, any solution candidate $S \subseteq Y'$ and any vertex $v \in V'$, v is observed in G after l rounds if and only if v is also observed in G' after l rounds of propagation. The vertices x_i and y_i are only observed after i rounds.*

Proof. We will prove this claim by induction on i.

Induction start $i = 1$. Since all additional vertices in G are not selectable, the application of the domination rule does not change for G. x_1 and y_1 are marked as pre-observed and thus observed after the first round. None of the other vertices x_j or y_j are pre-observed or connected to a selectable vertex, so they are not observed after the first round.

Induction step $i \rightarrow i+1$ Let $i < l-1$. The propagation rule can be applied to some vertex $u \in V'$ in G if and only if it can be applied in G' since $N[u]$ is unchanged for a target vertex u or gains one vertex v_u that is pre-observed and thus observed after the first round of propagation.

The propagation rule can also not be applied to the new vertices v_u since they are connected to x_{l-1} and y_{l-1} which are both not observed after round $i < l-1$.

The vertex x_{i+1} is connected to x_i which is observed and does not have other unobserved neighbors. The propagation rule can be applied to x_i to observe x_{i+1}. For any $j > i+1$, x_j is connected to x_{j-1} and to x_{j+1} if $j < l-1$ or to z_u if $j = l-1$. The vertices x_{j+1} and x_{j-1} are not observed after round i by the induction claim and cannot propagate. The vertex z_u cannot propagate as shown above. That means that x_j cannot become observed in round $i+1$. The same argument holds for y_{i+1} and y_j. ∎

Lemma 3.10: *Let $(G' = (V', E'), W' = V', X', Y', Z', T')$ be an instance of l-round EPDS with only propagating vertices and let $(G = (V, E), W = V, X', Y', Z, T = V)$ be the instance obtained by Construction 3.8. Then G' has a solution of size k if and only if G has a solution of size k for any $k \in \mathbb{N}$.*

Proof. We prove this lemma by proving that any set of vertices is a solution for G' if and only if it is a solution for G. First, we note that valid solution candidates are the same for G and G', since the construction does not change the selectable vertices. Now assume that we selected some set of vertices $S \subseteq Y'$.

The set of vertices $v \in V'$ that can be observed after $l-1$ rounds of parallel propagation is the same for G' and G. The target vertices of G' are not connected to any new vertices, so they can only become observed if the propagation rule is applied to some other vertex in $v \in V'$. Since the neighborhood of v' is either the same for G' and G or increases by one pre-observed vertex, the propagation rule can be applied the same in G as in G'. Thus, a target vertex of G' can be observed after l rounds in G if and only if it could also be observed after l rounds in G'.

All vertices $u \in V'$ that are not target vertices are connected to a new vertex z_u that is connected to x_{l-1} and y_{l-1}. Both x_{l-1} and y_{l-1} are observed after $l-1$ rounds of parallel propagation, so z_u can propagate in round l to observe u. Thus, all non-target vertices of G' are observed in G after l rounds. All new vertices of G are also observed after l rounds since they are either pre-observed or part of the paths that are observed after $l-1$ rounds.

In conclusion, every vertex of G is observed after l rounds of parallel propagation if and only if each target vertex of G' is observed after l rounds. ∎

3.2.3. Normalize pre-observed vertices

For any instance of l-round EPDS with pre-observed vertices, we provide an equivalent instance without pre-observed vertices without adding vertices that are not propagating or not a target vertex. We first add a pre-selected vertex x. We then add an edge from x to every pre-observed vertex $v \in Z$ and make v not pre-observed.

This reduction can be performed in $O(n)$ and increases the size of every solution by 1.

Lemma 3.11: *Let $(G' = (V', E'), W' = V', X', Y', Z', T' = V')$ be an instance of l-round EPDS and let $(G = (V, E), W = V, X, Y, Z = \emptyset, T = V)$ be the transformed instance. Then G' has a solution of size k if and only if G has a solution of size $k + 1$ for any $k \in \mathbb{N}$.*

Proof. Let S' be a solution of size k to l-round EPDS on G'. Then the set of vertices observed after the first round of propagation is $N[S'] \cup Z$. If we use $S = S' \cup \{x\}$ as a solution candidate to l-round EPDS on G, the set of vertices observed after the first round is $N[S] = N[S'] \cup N[x] = N[S'] \cup Z$. We see that the set of vertices observed after the first round is exactly the same, so S observes all vertices in G if and only S' observes all vertices in G'.

If S is a solution of size $k + 1$ for G, we know that $x \in S$. Thus, $S' = S \backslash \{x\}$ has size k. The set of vertices observed in G' by selecting S' is once again the same as the set of vertices observed in G by selecting S, so S' is a solution for G'. ∎

3.2.4. Normalize pre-selected vertices

For every instance of l-round EPDS with pre-selected vertices we provide an equivalent instance without pre-selected vertices by adding two leaves x_1 and x_2 to every pre-selected vertex x. The leaves are not selectable. This construction can be done in $O(n)$ and does not add vertices that are not propagating, not target vertices or pre-observed. We will show that it does not change any solutions of the instance. The construction and proof are also adapted from [BG25].

Lemma 3.12: *Let $(G' = (V', E'), W' = V', X', Y', Z' = \emptyset, T' = V')$ be an instance of l-round EPDS and let $(G = (V, E), W = V, X = \emptyset, Y', Z = \emptyset, T = V)$ be the transformed instance. Then G' has a solution of size k if and only if G has a solution of size k for any $k \in \mathbb{N}$.*

Proof. Let S be a solution of size k for G'. Since the transformation does not change the selectable vertices, we S is also a valid solution candidate for G. We know $x \in S$ for every pre-selected vertex x, so all leaves added to G are observed after the first round of propagation. The domination rule is not affected by adding more neighbors to selected vertices, so all vertices that are observed in round 1 in G' are also observed in round 1 in G. The neighborhood of vertices that are not selected is the same for G' and G, so the application of the propagation rule is not affected. This means that S is also a solution for G

Let S be a solution for G of size k. Let $x \in X'$ be a vertex that is pre-selected in G' with added leaves x_1 and x_2. We know that the added leaves are not in S since they are not selectable. The leaves x_1 and x_2 can only become observed by propagation or domination from x. However, x can never propagate as long as x_1 and x_2 are not observed, so it must be selected. That means that S is also a valid solution candidate for G'. Since the only difference between G' and G are some leaves that are connected to selected vertices, the set of vertices that can be observed by selecting S is the same for G' and G. Thus, S is also a solution for G'. ∎

3.2.5. Normalize selectable vertices

For an instance of l-round EPDS with non-selectable vertices we provide an equivalent instance without non-selectable vertices without adding vertices that are not propagating, not a target vertex, pre-observed or pre-selected. The construction will be possible in $O(k \cdot n)$. It will be possible for the constructed instance to have a solution even if the initial instance does not have a solution. This cannot be fully prevented because selecting all vertices is a solution for any instance with only selectable vertices. However, we can ensure for a given $k \in \mathbb{N}$ that the constructed instance has a solution of size k if and only if the initial instance also has a solution of size k. The construction and proof are also adapted from [BG25].

Construction 3.13: *Let $(G' = (V', E'), W' = V', X' = \emptyset, Y', Z' = \emptyset, T' = V')$ be an instance of l-round EPDS with parameter k. If $k > |Y'|$ we return a graph of $k + 1$ isolated vertices.*

Otherwise, initialize a graph G with $k + 2$ copies $G^1, ..., G^{k+2}$ of G' and a clique C containing all vertices of Y'. Two vertices in the same copy G^i are connected if their counterparts in G' are connected. A vertex x^c in the clique is connected to a vertex y^i in a copy G^i if x is in the closed neighborhood of y in the original graph. The resulting instance is normalized.

Claim 3.14: *Let $S \subseteq V$ be a solution candidate of size $k \leq |Y'|$ for G and let $S' \subseteq Y'$ be the set that contains the counterparts in G' of the vertices in $S \cap C$. If there are two graph copies $G^i = (V^i, E^i)$ and $G^{i'} = (V^{i'}, E^{i'})$ with $V^i \cap S = V^{i'} \cap S = \emptyset$, the copies x^i and $x^{i'}$ of some vertex x' observed after j rounds of parallel propagation for solution candidate S if and only if $x' \in V'$ is observed after j rounds of parallel propagation for solution candidate S'.*

Proof. The application of the domination rule can observe the same vertices in G^i as in G'. If a vertex $x' \in V'$ is observed through the domination rule in the original graph, there must be some selected vertex y' with $x' \in N[y']$. Because y' is selectable, it has a counterpart in C. This counterpart is connected to the counterpart of x' in G^i, so x^i can be observed through the domination rule. Since there are no selected vertices in G^i and no edges between graph copies, there are no other possible applications of the domination rule.

The application of the propagating rule is also unaffected if the set of observed vertices in G^i is equivalent to the set of observed vertices in G' for some round i. If a vertex $x' \in V'$ can propagate to some vertex $y' \in V'$, then x^i can also propagate to y^i since the only neighbors of x^i that are different than in G' are vertices in the clique C and all vertices in C are observed by the domination rule. It is also not possible for a vertex in the clique to propagate to an unobserved vertex x^i because the vertex $x^{i'}$ would also not be observed. This means that there are no additional application of the propagation rule in G^i since the different graph copies are not directly connected.

By induction, this means that Claim 3.14 is true for any $1 \leq j \leq l$. ∎

Lemma 3.15: *Let $(G' = (V', E'), W' = V', X' = \emptyset, Y', Z' = \emptyset, T' = V')$ be an instance of l-round EPDS and let $(G = (V, E), W = V, X = \emptyset, Y' = V, Z = \emptyset, T = V)$ be the instance obtained by Construction 3.13. Then G' has a solution of size k if and only if G has a solution of size k for any $k \in \mathbb{N}$.*

Proof. If $k > |Y'|$, the initial instance does not have a solution of size k since any solution has to be a subset of Y'. In this case, the construction provides an instance of $k + 1$ isolated vertices that only has a solution of size $k + 1$.

Assume $k \leq |Y'|$. Claim 3.14 directly shows that any solution $S' \subseteq V'$ for G' of size k produces a solution for G by selecting the counterparts of S' in the clique C.

Let S be a solution of size k for G. Since G contains $k+2$ copies of G', there are some i, i' with $V^i \cap S = V^{i'} \cap S = \emptyset$. Since S is a solution, all vertices in G^i and $G^{i'}$ must be observed after l rounds of parallel propagation. We can obtain a valid solution candidate S' for G' by selecting the counterparts of vertices in $S \cap C$, since they are all selectable. Applying Claim 3.14 to G^i and $G^{i'}$ shows that selecting S' observes all the vertices in G', so S' is a solution of size $k' \leq k$ for G'. We can obtain a solution of size k by adding additional vertices of Y' to S', since selecting more vertices only increases the number of observed vertices. ∎

3.3. Reduction from l-round PDS to WSAT⁺$[2l]$

We have shown that l-round PDS and l-round EPDS have the same parametrized complexity and are $W[2l]$ hard. To show that l-round PDS is also in $W[2l]$, we provide a parametrized reduction from l-round PDS to WSAT⁺$[2l]$. For this, we provide a circuit that simulates the parallel propagation for l steps. Each or-layer contains a node for every vertex in the initial graph, and an or-node in a certain layer outputs true if the respective vertex is observed after the same number of rounds in the original graph. The and-layers contain nodes for each edge in the graph and indicate whether a vertex can propagate along that edge in a certain round. The input layer is also contains a node for every vertex in the original graph, so every solution candidate for the instance of l-round PDS directly translates to a solution candidate for the constructed instance of WSAT⁺$[2l]$.

3.3.1. Construction

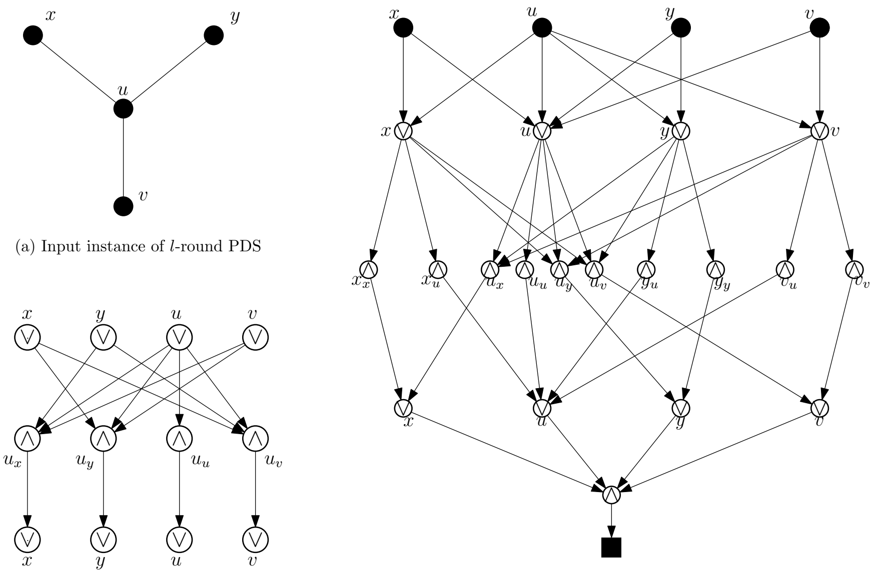
For an instance $G' = (V', E')$ of l-round PDS we define a circuit $G = (V, E)$ with $v = \bigcup_{i=0,\dots,2l+1} L_i$. For any parameter k, this provides an instance of WSAT+$[2l]$. We also define (partial) functions that map vertices in the input graph to nodes in the construction. The function $I : V' \to L_0$ maps each vertex to its respective input node, the function $O : V' \times \mathbb{N} \to \bigcup_{i=1,\dots,l} L_{2i-1}$ maps vertices to their or-nodes in a certain layer and the function $A : V' \times V' \times \mathbb{N} \to \bigcup_{i=1,\dots,l-1} L_{2i}$ maps pairs of adjacent vertices to the respective and-nodes.

Construction 3.16: *We add nodes to G as follows:*

- *Input layer: For every vertex $v' \in V'$ we add an input node v to L_0 with $I(v') = v$*
- *Odd layers: For every odd i with $1 \leq i \leq 2l-1$ we add an or-node v to L_i for every vertex $v' \in V'$ with $O(v', i) = v$*
- *Even layers: For every even i with $2 \leq i \leq 2l-2$, we add two vertices v_u and u_v for every edge $\{u', v'\} \in E'$ and a vertex x_x for every vertex $x' \in V'$. We define $A(u', v', i) = u_v$, $A(v', u', i) = v_u$ and $A(x', x', i) = x_x$.*
- *Last even layer: The layer L_{2l} consists of one and-node.*
- *Output layer: The output layer L_{2l+1} consists of one output node o.*

We add edges to G as follows:

- *First odd layer: An or-node $x \in L_1$ with $x = O(v', 1)$ has an incoming edge from an input node y with $y = I(u')$ if $u' = v'$ or $\{u', v'\} \in E$.*
- *Other odd layers: Every or-node $v \in L_i$ for an odd i with $3 \leq 2l-1$ with $v = O(v', i)$ has incoming edges from all and-nodes u_v with $u_v = A(u', v', i-1)$ and $v' \in N[u']$*



(a) Input instance of l-round PDS

(b) Propagation gadget for vertex u

(c) Resulting circuit for $l = 2$

Figure 3.3.: Example of a transformation from an instance of l-round PDS to a normalized circuit for $l = 2$. The input layer and all or-layers each contain a node for every vertex in the graph. For each and-layer and each vertex in the original graph, we add a propagation gadget that simulates the propagation from that vertex to other vertices.

- *Even layers: For every even i with $2 \leq i \leq 2l-2$, every and-node $u_v = A(u', v', i) \in L_i$ with $u' \neq v'$ has incoming edges from all vertices $x = O(x', i-1) \in L_{i-1}$ with $x' \in N[u'] \setminus \{v'\}$. Every $v_v = A(v', v', i) \in L_i$ has one incoming edge from $O(v', i-1)$.*
- *Last even layer: the layer L_{2l} consists of one and-node with incoming edges from all or-nodes in layer L_{2l-1}*
- *Output layer: layer L_{2l+1} consists of one output node with one incoming edge from the one and-node in layer L_{2i}*

3.3.2. Proof of equality

To prove that this construction correctly simulates l-round PDS for a graph G, we show that the or-layers simulate the observation of vertices in G for every solution candidate. Using this, we can show that the circuit outputs true for a solution candidate if and only if the entire graph would be observed after l rounds.

Claim 3.17: *Let $G' = (V', E')$ be an instance of l-round PDS and $G = (V, E)$ be the circuit obtained by Construction 3.16 with layers L_i. Let $v \in L_{2i-1}$ be an or-node with $v = O(v', 2i-1)$ for any i with $1 \leq i \leq l$ and let $S \subseteq V'$ be a solution candidate for G' with $X = I(S)$. Then v outputs true in the circuit when all input nodes in X are set to true exactly if v' can be observed after i rounds of parallel propagation with selected vertices S.*

Proof. We prove this claim by induction on i.

Induction start $i = 1$

In the first step of parallel propagation, the domination rule is applied to all vertices in S. A vertex $v' \in V'$ can be observed after the first step if it has an edge to a vertex $u' \in S$. In this case, $v = O(v', 1)$ has an incoming edge from $u = I(u') \in X$ and v outputs true. If v' is not observed after the first step, it does not have an edge to a vertex in S and v does not have an incoming edge from an input node in X, so v outputs false.

Induction step $i \to i + 1$ We assume claim Claim 3.17 to be true for some $i < l$

Let $v \in L_{2i+1}$ be an or-node with $v = O(v', 2i + 1)$. If v' is observed after $i + 1$ rounds of parallel propagation, then v' must be observed after i rounds or have a neighbor u' where $N[u']\backslash v'$ is observed after i rounds. If v' is observed after i rounds, $O(v', 2i - 1)$ must output true. Since v has an incoming edge from $v_v = A(v', v', 2i)$ and v_v has one incoming edge from $O(v', 2i - 1)$, v must also output true. If $v' \in N[u']$ with $N[u']\backslash v'$ observed after i rounds, v must have an incoming edge from an and-node $u_v = A(u, v, 2i)$ and u_v must have incoming edges from all or-nodes $x \in O(N[u']\backslash v', 2i - 1)$. From **??** we know that all or-nodes x output true, so u_v and v also output true.

If v' is not observed after $i+1$ rounds, then v' is not observed after i rounds and all neighbors of v' are not observed or have a neighbor other than v' that is not observed after i rounds. In the circuit, v only has incoming edges from v_v and from nodes $u_v = A(u', v', i)$ for all $u' \in N(v)$. v_v has an incoming edge from $O(v', 2i - 1)$, which outputs false, so v_v also outputs false. For any $u' \in N(v)$, u_v has incoming edges from all nodes $x \in O(N[u']\backslash v', 2i - 1)$. Since u' is not observed or has a neighbor other than v' that is not observed after round i, we know that one of input nodes x of u_v outputs false, so u_v also outputs false. Since all input nodes of v output false, v must also output false. ∎

Lemma 3.18: *Let $G' = (V', E')$ be an instance of l-round PDS, $S \subseteq V'$ a solution candidate of G' and $X = I(S)$ an assignment for the circuit G obtained by construction Construction 3.16. Then G (interpreted as a circuit) outputs true exactly if S is a solution of G'.*

Proof. Let G be the circuit obtained by applying construction Construction 3.16 to a graph $G' = (V', E')$ and $X = I(S)$ a solution candidate for G. The output node of G is connected to one and-node $v \in L_{2_l}$ that is connected to all nodes in layer L_{2l-1}. From Claim 3.17 we know that each node in L_{2l-1} outputs true for input set X if its counterpart in G' is observed after l rounds of parallel propagation for selected vertices S. Since the layer L_{2l} contains an and-node that has every node in L_{2l-1} as input, the circuit outputs true if and only if every vertex is observed after l rounds. Therefore, G outputs true for input X if and only if S is a solution for G' ∎

Corollary 3.19: *There is a parametrized reduction from l-round PDS to WSAT⁺[2l] when we parameterize l-round PDS.*

Proof. For any instance (G', k) of l-round PDS, construction Construction 3.16 provides an instance (G, k) of WSAT⁺$[2l]$. We have shown with Lemma 3.18 that every solution of size k for G' directly translates to a satisfying assignment of size k for G. The constructed graph is a valid instance of WSAT⁺$[2l]$ since it consists of $2l$ alternating layers of or-nodes and and-nodes, all and-nodes have output degree 1 and there are no edges that skip a layer. The construction can be computed in $O(l * n^2) = (n^2)$ since we perform the reduction for a constant l. ∎

4. Solving l-round Power Dominating Set

In this chapter we provide an algorithm to solve l-round EPDS. The algorithm works in two phases. In the first phase, we apply a set of reduction rules that each simplify the instance by either reducing the size of the graph, adding pre-selected vertices or reducing the number of selectable vertices. In the second phase, we apply the implicit hitting set approach based on [BFH18 | BG25] to l-round EPDS to optimally solve the instance. For simplicity, we only consider instances of l-round EPDS without pre-observed vertices. Note that for instances with pre-observed instances we can easily find an equivalent instance according to Section 3.2.3 without significantly increasing the complexity of the instance.

The two phases are modular and can also be combined with other algorithms. In Section 4.4 we combine our reduction rules with existing solvers for PDS by Jovanovic et al.[JV20] and Brimkov et al.[BMS19] and a solver for l-round PDS by Aazami [Aaz10].

4.1. Observation paths

For our reduction rules as well as the implicit hitting set approach, it is crucial to understand the impact that selecting a vertex can have on the observation of other vertices. Selecting a vertex v can only influence a limited number of vertices. That influence is limited by the parameter l and the preselected vertices. We introduce *observation paths* and *potential observation paths* to find limits on the influence of vertices.

Definition 4.1: *Let $(G = (V, E), W, X, Y, T)$ be an instance of extended l-round EPDS with selectable vertices $Y \subseteq V$ and preselected vertices $X \subseteq V$. Let $v \in Y \backslash X$ be a selectable vertex and $p = (x_1, ..., x_i)$ be a list of vertices. p is called an observation path for v if:*

- *x_1 is a neighbor of v*
- *Every vertex x_i becomes observed in round i when selecting $X \cup \{v\}$ but is not observed in round i when selecting only X.*
- *For every $2 \leq j \leq i$: x_i becomes observed either by propagation from x_{i-1} or from some vertex $z \in N[x_i] \cap N[x_{i-1}]$ where x_{i-1} is the last vertex to become observed in $N[z] \backslash \{x_i\}$.*

Lemma 4.2: *Let $(G = (V, E), W, X, Y, T)$ be an instance of l-round EPDS. For any vertex $v \in Y$ and any vertex $w \in V$: If selecting v changes the round in which w is observed, there must be an observation path starting at a neighbor of v and ending in w.*

Proof. The lemma can be shown by induction on the round i in which w becomes observed. If w becomes observed in round 1, it must be a neighbor of v and $p = (w)$ is an observation path.

If w becomes observed in a round $i > 1$, there must be some vertex z that propagates to w. If z becomes observed in round $i - 1$ by selecting v we know by induction that there must be an observation path from a neighbor of v to z. Otherwise, z must have a neighbor x that becomes observed in round $i - 1$ and becomes observed earlier by selecting v, since z could

otherwise propagate in round $i-1$ even if v was not selected. In either case, we know by induction that there is an observation path $p = (x_1, ..., x_{i-1})$ from a neighbor of v to either z or $x.p' = (x_1, ..., x_{i-1}, w)$ will still be an observation path as all requirements for an observation path hold for w. ∎

Observation paths are only defined for a given set of pre-selected vertices and provide little information about the influence of vertices if additional vertices are selected. For that, we give a broader definition of *potential observation paths*.

Definition 4.3: *Let $(G = (V, E), W, X, Y, T)$ be an instance of l-round EPDS and $p = (x_1, ..., x_i)$ a list of vertices. The list p is called a potential observation path if:*

- *The length of p is at most l*
- *x_j is not observed in round j for every $1 \leq j \leq l$*
- *For every $2 \leq j \leq i$: Either x_{i-1} and x_i are neighbors and x_{i-1} is propagating, or x_{i-1} and x_i have a common propagating neighbor.*

Lemma 4.4: *Let $(G = (V, E), W, X, Y, T)$ be an instance of l-round EPDS. Let $v \notin X$ be a vertex and $p = (x_1, ..., x_i)$ an observation path for v. Then p is a potential observation path.*

Proof. When selecting v in addition to X, every vertex in x_j in p becomes observed in round j, so p cannot be longer than l. The vertex x_j is also not observed in round j when selecting only X. For $j > 2$, the vertex x_j becomes observed either by propagation from its predecessor or from a vertex $z \in N[x_i] \cap N[x_{i-1}]$. That vertex must be propagating, which satisfies the last requirement for a potential observation path. ∎

Lemma 4.5: *Let $(G = (V, E), W, X, Y, T)$ be an instance of l-round EPDS with pre-selected vertices X and $p = (x_1, ..., x_i)$ a potential observation path. Then p is a potential observation path for any subset $X' \subseteq X$ of pre-selected vertices.*

Proof. The first and third requirements of potential observation paths are not impacted by changing the set of preselected vertices. The only requirement that could be affected is the round in which the vertices are observed. However, reducing the number of pre-selected vertices can only increase the round in which vertices become observed, so the second requirement also remains valid. ∎

Corollary 4.6: *Let $(G = (V, E), W, X, Y, T)$ be an instance of l-round EPDS with pre-selected vertices X and let v, w be vertices of G. If there is no potential observation path from a neighbor of v to w, then selecting v in addition to any superset $X' \supseteq X$ cannot change the round in which w becomes observed.*

Proof. Let $X' \supseteq X$ be a set of selected vertices. Assume that selecting v in addition to X' changes the observation time of w. From Lemma 4.2 we know that there must be an observation path p from a neighbor of v to w in the instance (G, W, X', Y, T). From Lemma 4.4 and Lemma 4.5 we know that p is also a potential observation path in the instance (G, W, X, Y, T) pre-selected vertices X. ∎

4.2. Reduction rules

Bläsius et al. [BG25] introduce reduction rules for an extension variant of regular Power Dominating Set that simplify a given graph without changing the optimal solution. More precisely, there is an equivalent solution after applying a rule for every optimal solution for the initial graph. Some of these rules can still be applied for l-round EPDS, while others have to be modified or are no longer valid. In this section, we will present rules that still work or can be modified, as well as introduce new reduction rules. Whenever we use or modify a rule introduced by Bläsius et al., we also use the same name for the rule.

For the following rules, we always assume an instance of l-round EPDS with a graph $G = (V, E)$, selectable vertices Y, pre-selected vertices X, propagating vertices W and target vertices T. We call a vertex *undecided* if it is selectable but not pre-selected.

The first two rules can be applied in the same way for extension variants of regular PDS or l-round PDS, and a are almost identical to the rules introduced by Bläsius et al. However, we slightly changed the rules to account for the addition of target vertices in our extension variant.

NecN Let $v \in Y \setminus X$ be an undecided vertex. If selecting all selectable vertices except v does not observe all target vertices in l rounds, then pre-select v.

Proof. Since selecting every selectable vertex except v does not observe all target vertices, v must be included in any possible solution. This means that the optimal solution remains the same when pre-selecting v. ∎

Isol Let $v \in Y$ be a selectable target vertex of degree 0. Then pre-select v.

Proof. The only way v can become observed is if v is selected. Since v is a target vertex, it must be included in every solution for G. ∎

The next rules are also very similar to rules introduced by Bläsius et al. and work almost the same for extension variants of l-round PDS and regular PDS. However, we need to be more careful with removing disabled vertices for l-round EPDS as this could change the number of rounds required to observe a graph.

Deg1a Let v be an undecided leaf connected to a selectable vertex w. Then v can be disabled.

Proof. Let $X' \supseteq X$ be an optimal solution for G. Assume X' includes v. If w is selected instead of v, the set of vertices observed after the first round could only increase because w is the only neighbor of v. This means that $X' \cup \{w\} \setminus \{v\}$ is still an optimal solution for G. ∎

Deg1b Let v be a disabled leaf connected to a selectable vertex w. If w is propagating and v is already observed or is not a target vertex, w is set to non-propagating. If w is not propagating and v is a target vertex, w is pre-selected. The leaf v is removed in both cases. If w is not propagating and v is not a target vertex, v is also removed.

Figure 4.1.: Example application of the rule DomLocal. If C contains a target vertex (left), v is pre-selected. If C does not contain a target vertex (right), v is set to non-propagating. C is removed in both cases.

Proof. First, assume that w is propagating and v is observed or not a target vertex. Since v is a leaf, it can only be observed by propagation or domination from w. This means that w cannot propagate to any vertex other than v since v only becomes observed after all other neighbors of w are observed. Thus, removing v and making w non-propagation does not change the behavior of w towards other vertices. Since v is either not a target vertex or already observed in the initial instance, removing it does not change the optimal solution.

Assume w is non-propagating. Removing v does not change the observation of any vertex other than v, as it cannot be selected and is only connected to w. It also cannot change whether w can propagate since w is not propagating anyway. If v is not a target vertex, it makes no difference whether v is observed, and it can be removed without changing the optimal solution. If v is a target vertex, the only way it can be observed is by domination from w. Thus, any optimal solution has to include w so it can be pre-selected. If w is pre-selected, v is observed after the first round and can be removed without changing the optimal solution. ∎

DomLocal[1] Let $C \subset V$ be a set of vertices with $|C| > 1$ and $v \in Y$ a selectable vertex that is not in C. If $C \subset N[v]$, remove C and set v to non-propagating. If C contains a target vertex, pre-select v.

Proof. First, we show that C can be safely excluded. Assume that there is an optimal solution that includes a vertex $x \in C$. Selecting a vertex in C only observes vertices in C and v with the domination rule. This means that selecting v observes at least the same vertices as selecting x, so $C \setminus \{x\} \cup \{v\}$ is still an optimal solution. Thus, x can be disabled.

If all vertices in $|C|$ are disabled, the only way they can be observed is by domination or propagation from v. Since $|C| > 1$, v will have at least two unobserved neighbors until a vertex in C becomes observed and therefore cannot propagate to C or any other vertex, so setting v to non-propagating does not change the application of the propagation rule in any round.

If there is a target vertex $x \in C$ that vertex has to be observed. The only way this is possible is if v is selected, so v can be pre-selected.

Now all vertices in C are disabled and only connected to each other or to v and v is non-propagating. This means that removing C does not change any observations outside of C as the propagation rule can never be applied to v anyway. If C contains a target vertex, we pre-selected v to ensure that it is observed in the initial instance. Otherwise, it does not matter whether C becomes observed. In either case, removing C does not change the optimal solution of G. ∎

[1] This rule is a more general version of rule Tri by Bläsius et al.

The following rules are also inspired by rules presented by Bläsius et al. and are in many cases very similar. However, their use is more restricted because changing the round in which vertices become observed can have a stronger impact for l-round PDS than for regular PDS.

Dom Let $v, w \in Y \backslash X$ be two different undecided vertices. If there is no potential observation path from a neighbor of w to an unobserved target vertex when selecting $X' = X \cup \{v\}$, w can be excluded.

Proof. Assume that $X' \supseteq X$ is an optimal solution for G that includes w. Then $X' \cup \{v\}$ is still a solution, as selecting more vertices can only cause vertices to become observed sooner. $X'' = X' \cup \{v\} \backslash \{w\}$ is a superset of $X \cup \{v\}$ since X does not contain w. We know from Corollary 4.6 that selecting w in addition to X'' will not change the observation time of any unobserved target vertex. Since selecting w does not observe additional target vertices, X'' must also be solution for G. Since $|X''| = |X'|$, X'' is an optimal solution for G that does not contain w. ∎

ObsE Let $v \in V \backslash Y$ be an excluded vertex that has no potential observation path to any unobserved target vertex. Let $w \in X$ be a preselected vertex. Then we can add an edge $\{v, w\}$ to G.

Proof. To show that this rule does not change the optimal solution, we show that the set of target vertices observed after l rounds does not change for any set $X' \supseteq X$ of selected vertices. It is easy to see that adding the edge does not cause any vertex to become observed later. That would only be possible if either v or w could not propagate in some round i because the other vertex is not observed. However, both v and w are observed in round 1 since w is pre-selected. Assume that adding the edge causes a vertex x to become observed earlier. Similar to Lemma 4.2 about selecting an additional vertex, this would require an observation path from to x from a vertex that becomes newly observed in the first round. Since v is the only vertex that becomes newly observed in the first round by this change, there would have to be an observation path from v to x. This means that x was already observed before adding the edge, or x is not a target vertex because there is no potential observation path from v to any unobserved target vertex. ∎

Round1 Let $\{v, w\} \in E$ be an edge and $v, w \in V \backslash X$ be unselected vertices that are in the neighborhood of pre-selected vertices. Then we can remove the edge $\{v, w\}$

Proof. We prove that removing the edge does not change the observation time of any vertex. The only way removing the edge would observe a vertex faster is if it prevents v or w from propagating because the other is not observed. Since both v and w are observed after the first round, this is not possible. The only way removing the edge would observe a vertex later is if either v or w dominates or propagates to the other. However, both vertices are neighbors to an unrelated pre-selected vertex, so they are both observed after the first round anyway. ∎

Deg2c Let $v \in N(X)$ be a propagating vertex that is observed after the first round of propagation. If v has precisely two neighbors x, y that are not observed after the first round of propagation, both are propagating and have degrees 2, and v, x and y are not selectable, we can remove the edges xv and vy and add an edge xy.

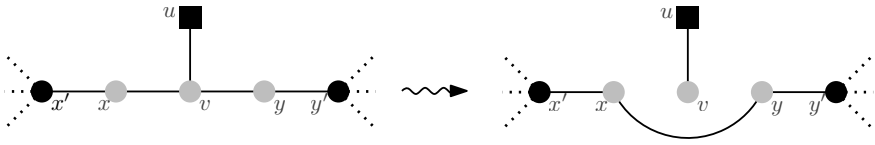


Figure 4.2.: Example application of rule Deg2c. If the vertices x and y of degree 2 are connected over a vertex that is observed in the first round, they can also be connected directly.

Proof. To prove that this rule does not change the minimal solution, we show that any application of the domination rule or propagation rule is equivalent for the initial and modified instance if the initially observed vertices are the same. This inductively proves that the set of vertices observed after l rounds are the same for both instances for any solution candidate.

Since only edges between v, x and y are changed and these vertices are all excluded, the application of the domination rule does not change. If the propagation rule is applied to any vertex other than x, y or v it can be applied the same way in both instances.

In the modified graph, the propagation rule can never be applied to the vertex v as all of its other neighbors are observed by round 1. In the initial graph, v can propagate to x or y. Assume that v propagates to y in the original graph. This is only possible if x is observed. The only way for x to become observed if y is not observed is by propagation from its other neighbor x'. If x' and x are both observed, x can propagate to y in the modified graph which provides an equivalent propagation.

Assume x propagates to its other neighbor x' in either graph. In the original graph, this is only possible if x became observed by propagation from v, which is only possible if y is already observed. In the modified graph, this is possible if x became observed by propagation from y, so y also has to be observed. Since the requirements for this propagation are the same in both instances, the application of the propagation rule does not change.

Lastly, assume that x propagates to a vertex other than x'. In the original instance, this is not possible as v is already observed after the first round. In the modified instance, x can propagate to y. However if x is observed in the original instance, v can propagate to y which again provides an equivalent propagation.

We do not need to give the same reasoning for y as x and y are symmetrical. ∎

The remaining rules are newly introduced by us. Rules Path1 and Path2 are only valid for l-round PDS.

Path1 Let $v_0, ..., v_k$ be a path. Let v_j be unobserved and a target vertex for some $0 \leq j \leq k$ and let v_i be observed or not a target vertex for all $i < j$. Let v_i be of degree 2, propagating and not preselected for $0 < i < k, i \neq j$ and let v_0 be either a leaf or preselected.

Case 1: $k \geq j + l$. Let v_p be the last selectable vertex in $v_0, ..., v_{j+l}$. Then we pre-select v_p and make all undecided vertices in $v_0, ..., v_{p-1}$ not selectable.

Case 2: $k < j + l$. Let v_p be the last selectable vertex on the path. Then we exclude all vertices in $v_0, ..., v_{p-1}$ and leave v_p undecided.

Proof. Let v_p be the last selectable vertex. If v_p is selected, then all vertices in $v_0, ..., v_p$ are observed or not target vertices. This is true by definition for the vertices in $v_0, ..., v_{j-1}$. If $p \geq j$, the vertices in $v_j, ..., v_p$ become observed by propagation along the path starting from v_p. The vertices in $v_0, ..., v_p$ do not have a potential observation path to any vertex outside that path. Otherwise, that path would have to include a vertex in the neighborhood of a selected

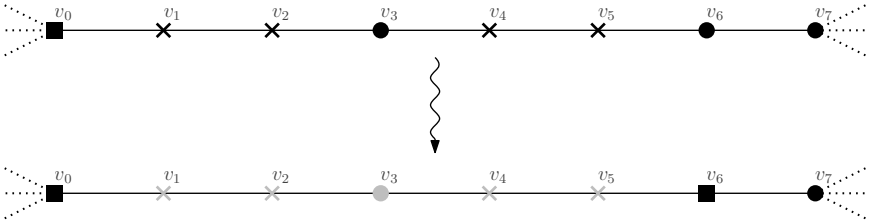


Figure 4.3.: Example application of rule Path1 for $l = 2$. The vertices v_1 and v_2 are observed by propagation starting from v_0. The first unobserved target vertex is v_4, so the vertex v_6 has to be pre-selected. The vertices $v_1, ..., v_5$ are excluded.

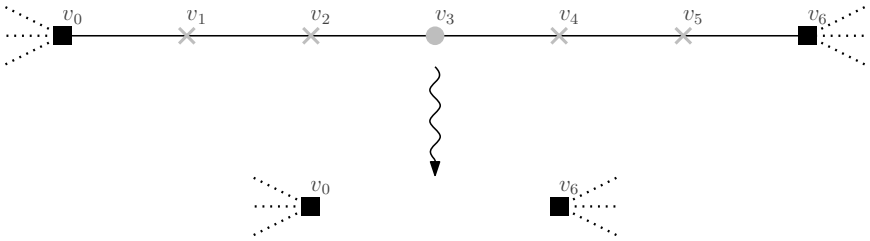


Figure 4.4.: Example application of rule Path2. All target vertices on the path $v_1, ..., v_5$ are observed and the path cannot influence the observation of the rest of the graph, so it can be removed.

vertex or skip at least three vertices. The first case would violate the second requirement for a potential observation path as a vertex on a potential observation path cannot be observed in the first round. The second case would violate the third requirement, as subsequent vertices on an observation path must be neighbors or have a common neighbor. According to rule Dom, the vertices $v_0, ..., v_{p-1}$ can thus be excluded.

If $k \geq j+l$, the only way for v_j to become observed is by propagation starting from a vertex in $v_0, ..., v_{j+l}$. Since v_p is the only remaining selectable vertex in $v_0, ..., v_{j+l}$ and v_j is a target vertex, v_p can be pre-selected.

If $k \geq j+l$, v_j also has to become observed by propagation over the path, but the propagation could still reach v_j if v_k becomes observed in a later round. This means that we cannot pre-select a vertex. ∎

Path2 Let $v_0, ..., v_k$ be a path where v_0 and v_k are pre-selected and v_i is of degree 2 and either observed or not a target vertex for $1 < i < k$. Then we can remove $v1, ..., vk - 1$.

Proof. The vertices $v_1, ..., v_{k-1}$ are only connected to other vertices of the path. The only vertices connected to the remaining graph are v_0 and v_k. Since they are pre-selected and all their neighbors are observed in round 1, it does not change their behavior if some of their neighbors are removed. This means that removing the vertices on the path can not change the observation time of any vertex outside the path. Since the path does not contain any target vertex that is not observed, removing the vertices does not change any solution for the instance. ∎

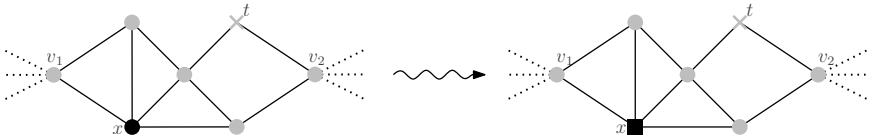


Figure 4.5.: Example application of rule NecLocal. The only way the target vertex t can become observed is if x is selected, so x can be pre-selected.

NecLocal Let C be a component in the graph obtained by removing all cut vertices from G and let $v_1, ..., v_k$ be the cut vertices connected to C. If x is the only selectable vertex in $C \cup \{v_1, ..., v_k\}$ and C contains at least one target vertex, then preselect x.

Proof. We can assume that every cut vertex v_i connected to C has at least two edges into C. Otherwise, the single vertex connected to v_i would also be a cut vertex and not a part of C. This means that it is impossible to propagate into C from the outside because no cut vertex can propagate as long as no vertex in C is observed. The only way to observe vertices in C is by selecting a vertex in the neighborhood of C. Since x is the only selectable vertex, it has to be selected to observe the target vertices in C and can therefore be pre-selected. ∎

4.3. Implicit Hitting Set Approach

In this section, we will adapt the implicit hitting set algorithm introduced by Brimkov et al. [BFH18] for solving regular Power Dominating Set.

In a graph $G = (V, E)$ a *fort* is a set of vertices $F \subseteq V$ such that there is no propagating vertex outside of F that has exactly one edge into F. These forts block propagation because no vertex can propagate to a vertex in F as long as the vertices in F are not observed. For regular EPDS, a set of vertices $S \subseteq V$ is a solution for G if and only if it is a hitting set in the set of forts in G that contain at least one target vertex.

For l-round EPDS, a solution for G still needs to be a hitting set in the sets of forts that contain a target vertex, as it is still impossible to propagate into a fort from an outside vertex. However, this condition is no longer sufficient, as propagating to a vertex might be possible but would take more than l rounds. Instead, we use the more general concept of *base sets*.

Definition 4.7: *Let $(G = (V, E), W, X, Y, T)$ be an instance of l-round EPDS and $B \subseteq V$ be a set of vertices. If selecting all selectable vertices in the complement of B does not observe all the target vertices in G, then B is a base set of G.*

For l-round EPDS, a set of vertices is a solution for a graph G if and only if it is a hitting set in the base sets of G. It is easy to see that a solution S must be a hitting set for the base sets of G, as otherwise S would be a subset of the complement of a base set and therefore could not observe all the target vertices in the graph. And the complement of a set of vertices S that is not a solution is a base set, so S cannot hit all base sets of G. Using this definition, we can use the following implicit hitting set algorithm. Start with a set of base sets $\mathcal{B}$ and calculate a minimum hitting set S of $\mathcal{B}$. If S is a solution for G, it is also an optimal solution. Otherwise, find an additional base set, add it to $\mathcal{B}$ and repeat.

The challenge of this approach is to find the right base sets to add after each step. For a base set to be considered good, we have two requirements.

- The new base set B is restricting such that the current solution to $\mathcal{B}$ is not a solution to $\mathcal{B} \cup \{B\}$
- We want to maximize the chance that the new set B improves our current solution candidate by observing additional vertices.

In the following, we will provide some methods for generating base sets and discuss how well they match our requirements.

Fort The selectable vertices in the neighborhood B of a fort that contains at least one target vertex x are also a base set. It is impossible to propagate into a fort from the outside, so at least one vertex in F or adjacent to F must be selected to observe x. If there is a fully unobserved fort F that contains at least one target vertex after selecting the current hitting set, adding B can be a good option, as selecting any vertex in B observes at least part of the fort that was not observed in the previous solution. Note that this does not necessarily mean that the next hitting set will observe more vertices than the last. However, finding a fully unobserved fort is not always possible, and adding a base set for a fort might not be useful if parts of the fort are already observed.

l-neighborhood The selectable vertices in the l-neighborhood of a target vertex v are a base set. Since propagation can cover a distance of at most l, any solution of l-round PDS must also be a solution of l-Dominating Set. Using the l-neighborhood of some target vertices in the graph can be a good starting point, as they often have limited size and are easy to find. However, adding the k-neighborhood later in the algorithm is not very useful as it might not even be more restricting than the already existing base sets.

Observation paths If we have a set of vertices S that does not observe some target vertex v, we can select additional vertices according to Lemma 4.9 that will not observe v. Then the neighborhood B of all potential observation paths ending in v is a base set (see Lemma 4.8). This is the most flexible approach we have found. The new base set is locally limited and guaranteed to further restrict possible solutions. Although selecting a vertex in B does not always improve the current solution by observing additional vertices, it is at least limited to vertices that can potentially observe an additional target vertex.

Lemma 4.8: *Let $(G = (V, E), W, X, Y, T)$ be an instance of l-round EPDS, $S \subseteq Y$ a set of vertices that is not a solution for G and v a target vertex that is not observed by selecting S. Then the neighborhood B of all vertices that have a potential observation path to v is a base set.*

Proof. Let $S' = V \backslash B$ be the complement of B. No vertex in S' has a neighbor with a potential observation path to v. S' is a superset of S since a potential observation path cannot include the neighbor of a selected vertex. From Corollary 4.6 we know that selecting a vertex in S' does not change the round in which v is observed regardless of what other vertices are selected. That means that selecting S' does not observe v and is therefore not a solution for G. ∎

Lemma 4.9: *Let $(G = (V, E), W, X, Y, T)$ be an instance of l-round PDS, $S \subseteq Y$ a set of vertices that is not a solution for G and v a vertex that would only be observed after $k > l$ rounds of observation by selecting S. If we select every selectable vertex except for the neighborhood of vertices that are not observed by round $k - l$, then v will still not be observed by round l.*

Proof. Set $S' \supseteq S$ be the set of selectable vertices that are not in the neighborhood of a vertex not observed by round $k - l$. If we select S', all vertices observed after the first round would have been observed by round $k - l$ previously. After observing all vertices previously observed by round $k - l$ it would still take l more rounds to observe v. This means that v is not observed before round $l + 1$ by selecting S'. ∎

This leads to the following algorithm for finding base sets with the implicit hitting set approach. We initialize our base sets with the l-neighborhood of randomly or evenly distributed vertices. After finding a potential solution, we calculate the exhaustive propagation. If we find a set of vertices that are still unobserved, they form a fort. If that fort contains at least one target vertex, the selectable vertices in their neighborhood can be added as a base set. If this is not the case and we find a target vertex that becomes observed in a round $k > l$, we select all vertices except for the neighborhood of vertices not observed in round $k - l$, calculate the potential observation paths ending in v and add the neighborhood of these paths as a base set. This algorithm is correct, as we only add valid base sets to our hitting set instance. It is also complete because there is a finite set of base sets for any instance and we add new base sets in each iteration.

4.3.1. Implementation notes

Finding l-neighborhoods and forts for a set of selected vertices is relatively straightforward. However, we will describe our method for finding base sets based on observation paths. We start with a target vertex v that is not observed after l rounds. We extend our base set by exploring vertices in l rounds. To explore a vertex w, we first generate the 2-neighborhood of w. If a vertex x is propagating and a neighbor of w or x has a common propagating neighbor with w, we add x to the base set. In each round, we only explore the vertices that were added to the base set in the last round. Finally, we return the neighborhood of all added vertices as a base set. Exploring the 2-neighborhood of a vertex has a worst-case complexity of $O(m)$. Since we never explore a vertex more than once, the worst-case complexity of this method is $O(n \cdot m)$.

For the implicit hitting set approach, we generally need to balance the time spent finding new base sets with the time spent solving the hitting set instances. Finding multiple base sets in each iteration costs more time, but can reduce the number of hitting sets we have to solve. We therefore implemented and compared different methods for generating multiple base sets in each iteration.

l-neighborhoods For the initial base sets, we initially added the l-neighborhood of every target vertex that is not in the l-neighborhood of a preselected vertex as a base set. We hoped that this would produce a close estimate of a solution for small l. However, this method proved to be very inefficient for larger l. Our second attempt was iteratively adding the l-neighborhood of target vertices that are not in the l-neighborhood of a preselected vertex or in a previous base set. We hoped that this would provide a good spread of base sets while remaining more efficient for larger l. In the end, we tested adding base sets for all target vertices, adding base sets for limited target vertices, and adding no l-neighborhoods as base sets at all.

Forts For finding forts, we use the method presented by Bläsius et al. [BG25] to find inclusion-minimal forts. This method starts with a set of selected vertices that is not a solution for l-round EPDS and generates additional candidate solutions. For this, we have a randomly ordered list $L = v_1, ..., v_k$ of vertices that are currently not selected or excluded. We start by selecting all vertices in L and then iterate over the list. If all target vertices are observed after exhaustive propagation, we un-select the current vertex v_i. Otherwise, we re-select v_{i-1} and un-select v_i. Whenever we find a candidate solution that is not an actual solution, the unobserved vertices are a fort. We ensure that we add an inclusion-minimal fort containing a target vertex by iteratively re-selecting vertices in the fort neighborhood and test whether this still results in a fort containing a target vertex.

Additionally, we tested a method of greedily generating forts that works as follows. We start with a solution candidate S a set of unobserved vertices F and iterate over the vertices in $N[F]$. For every vertex x in $N[F]$, we select x. If this observes all target vertices in the graph, we un-select x and add it to an inclusion minimal base set B. If it does not observe all target vertices, we leave x selected and continue. Once we tested all vertices , we un-select all vertices in $N[F]$ and select one vertex in B. We repeat this process until all target vertices are observed.

Other base sets For finding the remaining base sets based on observation paths, we also tested different methods. For the first method, we select a random target vertex x that is not observed after l rounds of propagation. We find a base set of vertices that have neighbors with a potential observation path to x and then activate x. We repeat this until all target vertices are observed after l rounds.

The second method works similarly, except that we prioritize selecting target vertices that become observed in the latest rounds. We hope that this produces more useful base sets as they should cover more vertices that are not observed by round l.

Lastly, we use an adaptation of the method used by Bläsius et al. We generate additional candidate solutions by selecting and un-selecting vertices in the same way. For each candidate solution that is not an actual solution, we find base sets for the target vertices that become observed last.

Additionally, we test whether ensuring that base sets are inclusion-minimal is useful for these base sets. To find an inclusion minimal base set F for our initial base set F' that is required to observe a vertex x in round l, we first select all vertices surrounding F'. We then iterate over the vertices in F' and select them. If selecting a vertex $u \in F'$ observes x we add it to F and un-select it. Otherwise, we leave it selected. This method can be more time-consuming than the method for finding inclusion minimal forts, so it might be more efficient to use slightly less precise base sets that are easier to find.

4.4. Experiments

In this section, we present the practical results of our implementation. We compare the run-time of our solver to several state-of-the-art implementations. Additionally, we analyze the effect of our reduction rules on the solver, the effectiveness of different optimizations, and the performance of our solver based on the value for l.

There are different existing approaches to solve or approximate l-round PDS on special types of graphs. But to our knowledge, the only approach that is feasible in practice on arbitrary graphs is using an integer linear programming formulation (MILP). We therefore use different MILP formulations as a baseline for the state-of-the-art to evaluate our algorithm.

The first formulation we chose is by Aazami [Aaz10]. This is the only formulation we have found that is specifically designed for l-round PDS. However, we had to modify it slightly because the original formulation does not accept solutions that can observe the entire graph in less than l steps. We also chose a formulation by Brimkov et al. [BMS19]. To our knowledge is the only formulation that directly works for l-round PDS that has been evaluated in practice, even if Brimkov et al. only evaluated it for bounded propagation that is possible with a minimal regular Power Dominating Set. The last formulation we chose is for regular PDS and was provided by Jovanovic et al. [JV20]. However, it has been used in practice and can be used for regular PDS with relatively simple adaptations. Additionally, Bläsius et al [BG25] have already provided an adaptation of this formulation for an extension variant of PDS.

We refer to the first formulation as AAZ, the second formulation as BRIM and the last formulation as JOV. We also adapted the formulations AAZ and BRIM for our extension variant of l-round PDS, but we used the original formulation on normalized instances. We provide our adaptations of all MILP formulations in Appendix A.1

4.4.1. Experiment Setup

We implemented the reduction rules and the solver in C++ 20 and compiled it with Clang 18.1.3. Our source code along with the data sets, evaluation scripts and raw data is available on our GitLab [2]. The source code is based on a solver for regular Power Dominating Set by Max Göttlicher [3].

We use Gurobi 11.0.2 [25] to solve the MILP formulations as well as the hitting set instances for our own solver. The experiments were run on a machine running Ubuntu 24.04.2 with Linux 6.8.0. The machine has two Intel Xeon(tm) Skylake SP Gold 6144 CPUs with a frequency of 3.5-4.2 GHz and 192 GB RAM.

We tested the solver on the graphs shipped with pandapower [Thu+18], which we interpret as instances of l-round EPDS in the normalized form. We had to exclude the largest instance (case9241pegase) from most of our tests, as our solver took up to 30 minutes for some l, which would have stalled our experiments. For our solver, we generally used a timeout of 10 minutes, and for the MILP formulations we used a timeout of 15 min. We tested the instances over a range of different values for l. For most experiments, we tested $l = 1, 2, 4, 8, 15, 30, 60$. We did not test larger values as we never found any solution candidates that could power dominate the instances in more than 50 rounds, which makes l-round PDS for $l > 50$ identical to regular PDS. For our comparison of the performance based on l we tested $l = 1, ..., 50$ for more fine-grained results. For our comparison with the MILP solvers, we limited the tests to $l = 2, 8, 15$ since they were generally less efficient. Especially the formulation by Aazami was very slow for larger l, so testing larger l would have taken too long.

[2] https://gitlab.com/frog711/l-round-pds-code
[3] https://gitlab.com/Aldorn/pds-code

4.4.2. Performance Comparison

We tested our own algorithm and all MILPs by themselves and in combination with our reduction rules. We also tested the algorithms for different l to see how this would impact performance. Table 4.1, Table 4.2 and Table 4.3 show the run-time for for various l. If the solver is combined with the reduction rules, we provide the added time of the reduction rules and the solver itself. In some cases, our reduction rules were able to find a result by themselves. For these instances, the reduction rules were faster than any solver. In all other instances, our algorithm was faster than any of the MILP formulations. We can also see that in some instances, the run-time is longer for our algorithm when we use the reduction rules. However, most of these instances can be solved relatively fast anyway. For the majority of instances, and especially more complicated instances, the reduction rules significantly reduce the computation time. Interestingly, the reduction rules sometimes increased the run-time of the solver for the largest instance, even if the time spent on the reduction rules itself is neglected. For the MILP solvers, the reduction rules are almost always beneficial. Another interesting thing to note is that our algorithm generally becomes faster if l is close to the number of rounds required to observe the graph with the optimal solution to regular PDS (see Section 4.4.5 for more detail). This effect does not occur with any of the MILP formulations. Especially the solver provided by Aazami performs worse for larger l. This leads to a speed-up that is particularly high for larger l. While we have a median speedup of 4.00 for $l = 2$, we see a median speedup of 18.38 for $l = 15$. Furthermore, the highest speed-up achieved increases from 48.78 for $l = 2$ to 1397.89 for $l = 15$.

4.4.3. Comparison of reduction rules

We tested different configurations of reduction rules. We refer to all rules except for Dom and NecN as local rules as they only consider a local part of the graph. We refer to Dom and NecN as global rules as they consider the entire graph. Apart from running the solver by itself, we tested two configurations of reduction rules. For the local configuration, we exhaustively apply local reduction rules. For the global configuration, we alternately apply all local and all global reduction rules. We measure the speed-up achieved by both configurations compared to running the solver by itself. We repeated the experiment for $l = 1, 2, 4, 8, 15, 30, 60$. Figure 4.6 shows the results accumulated in a box plot. The speed-up is greater than one for 113 of 175 test runs for local and 117 of 175 test runs for global reduction rules. The median speed-up is very similar with 1.33 for local and 1.23 for global reduction rules. However, we see that the global reduction rules are primarily useful in very small instances that can often be solved optimally by reduction rules. For larger instances, we see a more reliable speedup when using only local reduction rules. Figure 4.7 shows the number of vertices selected or disabled and the time taken to use only local reduction rules or also using global reduction rules. Although adding global reduction rules almost doubles the number of selected vertices in many cases and also increases the number of disabled vertices, it mostly increases the time taken by the reduction rules. It appears that the time save of our solver achieved by the reduction rules is not enough to apply more expensive reduction rules even if they lead to better results.

In addition to analyzing the speed-up achieved with the reduction rules, we also investigated the effect that the reduction rules have on different parts of our solver. For this, we measured the number of forts and base sets based on observation paths that the solver needs to find an optimal solution. If the solver needs fewer base sets to find a solution, it also has to solve fewer base sets and should generally be faster. Figure 4.8 shows the reduction in the number

Table 4.1.: Comparison of different solvers and reduction rules for $l = 2$. Note that some run-times are given in ms

path case*	n #	AAZ s	AAZ+R s	JOV s	JOV+R s	BRIM s	BRIM+R s	Ours s	Ours+R s	Speedup[a)]
4gs[b)]	4	14m	3m	19m	3m	52m	3m	3m	3m	4.67
5[b)]	5	8m	3m	8m	3m	9m	3m	3m	3m	2.67
6ww[b)]	6	5m	3m	5m	3m	11m	3m	4m	3m	1.67
9	9	8m	7m	8m	7m	11m	14m	6m	5m	1.60
11_iwamoto	11	8m	6m	11m	5m	13m	11m	5m	5m	1.60
14	14	11m	7m	13m	9m	20m	17m	4m	4m	2.75
24_ieee_rts	24	30m	19m	58m	35m	203m	75m	12m	19m	1.58
30	30	20m	9m	21m	15m	65m	27m	6m	5m	4.00
_ieee30	30	17m	11m	22m	16m	87m	37m	5m	6m	2.83
33bw	33	14m	4m	16m	5m	58m	5m	6m	3m	4.67
39	39	16m	8m	21m	12m	180m	36m	11m	7m	2.29
57	57	85m	48m	234m	82m	487m	346m	20m	18m	4.72
89pegase	89	135m	39m	550m	78m	2.85	1.44	23m	33m	4.09
118	118	94m	41m	313m	100m	2.32	581m	26m	35m	2.69
145	145	295m	62m	573m	99m	5.37	949m	13m	13m	22.69
_illinois200	200	53m	20m	50m	28m	617m	96m	17m	14m	3.57
300	300	238m	61m	679m	125m	10.83	3.13	61m	62m	3.84
1354pegase	1354	332m	132m	428m	172m	12.72	1.27	163m	88m	3.77
1888rte	1888	414m	215m	720m	228m	15.96	417m	173m	94m	4.40
2848rte	2848	630m	401m	1.05	330m	21.97	962m	339m	127m	4.96
2869pegase	2869	43.83	14.62	80.61	31.09	>900	341.52	4.29	3.67	11.93
3120sp	3120	19.36	2.41	24.67	4.1	185.19	91.84	1.43	997m	19.42
6470rte	6470	5.51	1.11	13.28	1.17	226.45	11.67	1.22	713m	7.73
6495rte	6495	3.93	1.09	14.81	1.13	182.47	12.09	1.19	658m	5.98
6515rte	6515	7.19	1.13	6.96	1.13	235.13	11.62	1.03	732m	9.51
9241pegase	9241	>3600	>3600	>3600	>3600	>3600	>3600	51.07	73.8	>48.78

[a)] speedup of Ours+R compared to the better of JOV, AAZ and BRIM
[b)] solved optimally by reduction rules

of forts and base sets based on observation paths found by using all reduction rules compared to using none. While the reduction rules decrease the number of forts found for 156 of 175 test runs and by a median factor of 6, they only reduce the number of other base sets found for 69 of 175 test runs and by a median factor of 1.32. This suggests that reduction rules are more helpful for the solver while it finds an initial Power Dominating Set than while it finds and actual l-round Power Dominating Set.

4.4.4. Comparison of optimizations

In Section 4.3.1, we discuss different methods for generating base sets. For finding forts and other base sets, we always use the most simple greedy method as a baseline and measure the speed-up achieved by using more sophisticated methods. We also measure the speed-up achieved by minimizing base sets based on observation paths or adding the l-neighborhood of vertices as base sets. We always test the optimizations by themselves and in combination with our reduction rules. Figure 4.9 shows the speed-up of our solver achieved by activating multiple optimizations compared to using none. For calculating the speed-up, we only consider the run-time of the solver itself and ignore the run-time of the reduction rules as they are not impacted by the optimizations. For testing the solver by itself, we used the shuffle approach

Table 4.2.: Comparison of different solvers and reduction rules for $l = 8$. Note that some run-times are given in ms

path case*	n #	AAZ s	AAZ+R s	JOV s	JOV+R s	BRIM s	BRIM+R s	Ours s	Ours+R s	Speedup[a)]
4gs[b)]	4	8m	3m	9m	3m	9m	3m	3m	3m	2.67
5[b)]	5	10m	3m	10m	3m	9m	3m	4m	3m	3.00
6ww[b)]	6	10m	3m	10m	3m	12m	3m	3m	3m	3.33
9	9	37m	34m	11m	10m	11m	11m	4m	4m	2.75
11_iwamoto[b)]	11	30m	3m	10m	3m	10m	3m	4m	3m	3.33
14	14	131m	101m	31m	19m	48m	35m	4m	4m	7.75
24_ieee_rts	24	1.62	441m	190m	137m	120m	96m	4m	3m	40.00
_ieee30	30	1.15	424m	40m	33m	74m	40m	5m	4m	10.00
30	30	1.14	333m	43m	27m	172m	55m	4m	4m	10.75
33bw	33	285m	nan	35m	nan	20m	nan	4m	3m	6.67
39	39	1.66	98m	34m	20m	46m	23m	4m	4m	8.50
57	57	17.46	9.35	414m	414m	470m	737m	15m	11m	37.64
89pegase	89	11.45	10.85	2.53	1.19	4.08	1.41	12m	11m	230.18
118	118	417.07	113.65	2.02	725m	3.17	2.63	51m	100m	20.16
145	145	150.16	1.03	17.55	529m	14.04	14.76	9m	11m	1276.72
_illinois200	200	14.99	131m	79m	48m	258m	60m	14m	10m	7.90
300	300	>900	>900	32.45	41.49	64.61	29.49	102m	229m	141.71
1354pegase	1354	>900	11.48	749m	358m	4.08	733m	73m	60m	12.48
1888rte	1888	>900	12.72	8.88	522m	18.55	1.37	114m	89m	99.79
2848rte	2848	>900	41.04	7.6	1.42	9.4	2.31	190m	127m	59.85
2869pegase	2869	>900	>900	>900	>900	>900	>900	126.65	95.82	>9.40
3120sp	3120	>900	>900	>900	>900	>900	>900	291.0	287.7	>3.13
6470rte	6470	>900	>900	260.04	54.05	>900	197.77	1.06	675m	385.24
6495rte	6495	>900	>900	732.47	25.23	>900	98.16	1.11	678m	1080.34
6515rte	6515	>900	>900	591.95	25.17	>900	76.83	833m	535m	1106.45
9241pegase	9241	>3600	>3600	>3600	>3600	>3600	>3600	1411.77	289.43	>12.44

[a)] speedup of Ours+R compared to the better of JOV, AAZ and BRIM
[b)] solved optimally by reduction rules

to generate forts and base sets based on observation paths, minimized all base sets, and added sparse l-neighborhoods as base sets. For testing the solver in combination with reduction rules, we did not add any l-neighborhoods as we found that even the sparse approach was very time-consuming for large l when used in combination with the reduction rules due to an imperfect implementation.

We found that the optimizations generally give a speed-up of at least one and provide a significant speed-up for some instances when the solver is used by itself, but are less useful when the solver is used with reduction rules. An explanation for this could be the uneven impact of reduction rules on the solver performance. In Section 4.4.3 we discuss that the reduction rules reduce the number of forts found more than the number of base sets based on observation paths. We also found that optimizing fort generation brings the most reliable speedup compared to the other optimizations (See Figure A.1 for speedup achieved by individual optimizations). This means that the reduction rules reduce the impact of finding forts on solver performance, which is also the part of the algorithm where our optimizations are the most significant.

Table 4.3.: Comparison of different solvers and reduction rules for $l = 15$. Note that some run-times are given in ms

path case*	n #	AAZ s	AAZ+R s	JOV s	JOV+R s	BRIM s	BRIM+R s	Ours s	Ours+R s	Speedup[a)]
4gs[b)]	4	8m	2m	13m	2m	9m	2m	3m	2m	4.00
5[b)]	5	10m	2m	11m	2m	8m	2m	3m	2m	4.00
6ww[b)]	6	10m	2m	11m	2m	13m	2m	3m	2m	5.00
9	9	50m	33m	11m	11m	9m	11m	3m	3m	3.00
11_iwamoto[b)]	11	45m	2m	11m	2m	10m	2m	4m	2m	5.00
14	14	621m	351m	35m	17m	28m	29m	3m	3m	9.33
24_ieee_rts	24	3.62	3.95	225m	64m	125m	142m	3m	3m	41.67
30	30	6.29	2.29	41m	29m	101m	38m	4m	4m	10.25
_ieee30	30	6.67	1.93	48m	34m	87m	32m	4m	4m	12.00
33bw[b)]	33	1.53	2m	24m	2m	17m	2m	3m	2m	8.5
39	39	9.6	431m	35m	21m	57m	24m	3m	3m	11.67
57	57	210.8	133.69	298m	173m	306m	443m	22m	18m	16.56
89pegase	89	572.82	105.3	824m	564m	2.87	1.5	10m	7m	117.71
118	118	>900	>900	687m	480m	1.97	1.12	27m	26m	26.42
145	145	>900	3.25	16.25	784m	12.58	10.7	7m	9m	1397.89
_illinois200	200	235.43	375m	68m	31m	306m	25m	11m	8m	8.50
300	300	>900	>900	5.44	862m	16.56	6.52	220m	182m	29.88
1354pegase	1354	>900	191.36	864m	353m	3.56	797m	57m	47m	18.38
1888rte	1888	>900	133.46	3.95	502m	6.7	1.03	98m	63m	62.75
2848rte	2848	>900	322.58	12.84	1.44	7.88	2.28	147m	122m	64.56
2869pegase	2869	>900	>900	>900	>900	>900	>900	6.55	19.71	>45.67
3120sp	3120	>900	>900	>900	>900	>900	>900	80.11	48.44	>18.59
6470rte	6470	>900	>900	507.31	54.03	>900	68.99	734m	562m	902.69
6495rte	6495	>900	>900	261.23	8.11	765.53	29.62	666m	555m	470.69
6515rte	6515	>900	>900	383.12	5.92	877.49	18.87	625m	559m	685.37
9241pegase	9241	>3600	>3600	>3600	>3600	>3600	>3600	260.03	252.41	>14.26

[a)] speedup of Ours+R compared to the better of JOV, AAZ and BRIM
[b)] solved optimally by reduction rules

4.4.5. Performance by l

The last thing we were interested in is the performance of our solver based on the parameter l. We used our algorithm with the shuffle variant for finding both forts and base sets based on observation paths and minimized all base sets. Additionally, we used all our reduction rules to use our solver with optimal settings. We tested the performance of the solver and reduction rules for $l = 1, ..., 50$.

For the reduction rules, we generally found a lower runtime and number of disabled vertices for very small l and a higher number of selected vertices for l=1. Apart from that, the performance of reduction rules was very stable for different l. The reason for the higher performance for small l is that potential observation paths and propagation paths can become longer for larger l.

For the solver itself, we measure the runtime and the total number of forts and other base sets found. We excluded some of the smallest instances from our analysis that were optimally solved by reduction rules. Figure 4.10 shows the solver performance for different instances with different numbers of vertices along with the number of rounds to observe the graph for a regular minimal Power Dominating Set. We show this for the minimal Power Dominating Set that can observe the graph the fastest and for the Power Dominating Set that our solver

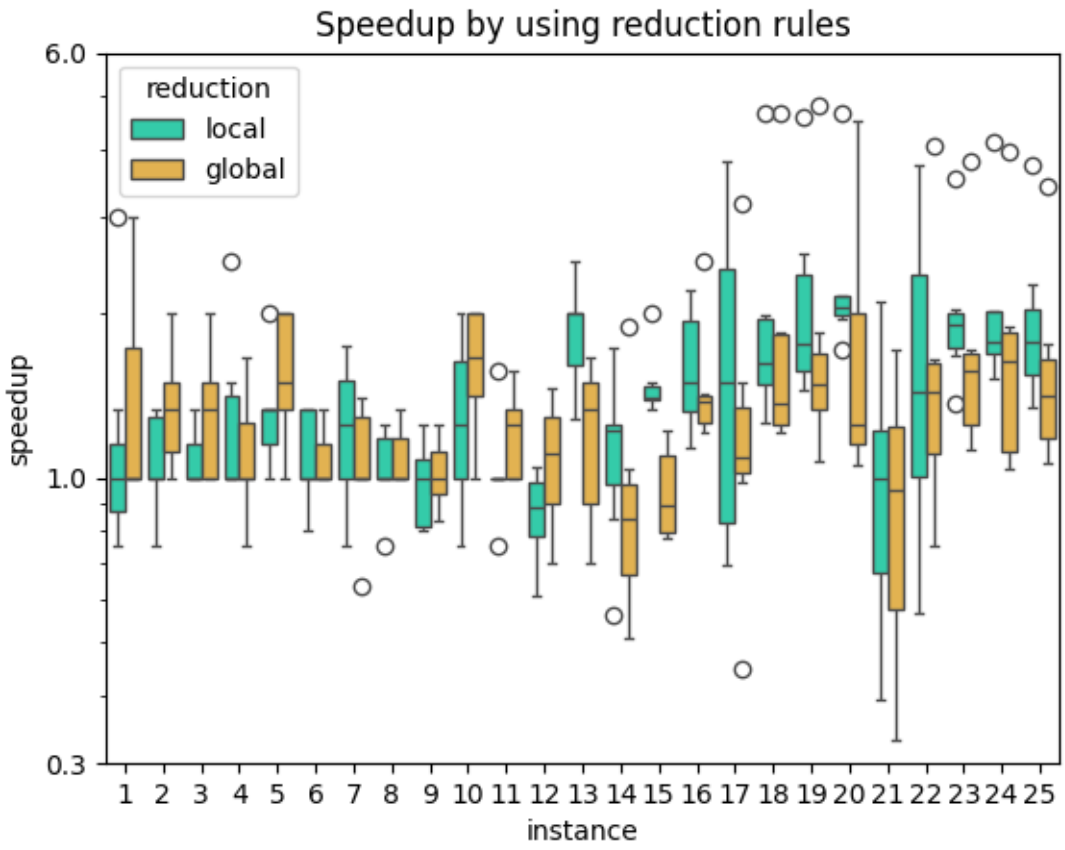


Figure 4.6.: Speedup of time taken by reduction rules and solver to time taken by using the solver by itself. The speedup is aggregated over different l for each instance. For *local* we only used local reduction rules, for *global* we used all of our reduction rules. The instances are ordered by number of vertices.

found without restrictions on l. Note that the solver does not always find a l-round Power Dominating Set by using only forts even if it would be possible because it can find a different Power Dominating Set that requires more rounds. We noticed that the number of rounds is often very similar for both cases, which suggests that the number of rounds required to observe the graph is actually relatively similar for a large share of minimal Power Dominating Sets.

The number of forts found is generally very stable, apart from a lower number for small l. This is an expected result since we use exhaustive propagation to find forts, so the algorithm does not depend on l until it finds an initial solution to Power Dominating Set. Although the performance of reduction rules can change depending on the value for l and it is also possible that the algorithm has to add additional forts after finding an initial solution to PDS, it appears that these do not have a significant impact on the solver. The runtime of the solver appears to be correlated with the number of base sets based on observation paths. Both rise sharply for small l and have a main peak between $l = 3$ and $l = 10$, with the peak of the runtime sometimes having a small offset to the right. Our explanation for this rise is that for larger l, the potential observation paths become longer, and the base sets become larger. As base sets

are added to help observe certain vertices, larger base sets make it less likely that selecting one of their vertices will observe the right vertex. Additionally, finding and minimizing larger base sets takes more time, so the runtime is impacted by larger l even more than the number of base sets required.

After the initial peak, both the runtime and the number of base sets based on observation paths quickly decline. We assume that for larger l, the Power Dominating Set found by just using forts as base sets is already close to a l-round Power Dominating Set, so it only requires few additional base sets to solve the instance. We also found that for some of the larger instances, even small changes in the number of base sets based on observation paths can correlate to large changes in runtime. Figure 4.11 shows the runtime of the solver compared to the log of number of base sets for two of our largest instances. This suggests that for large l, even finding only a small number of base sets might cost a lot of time. Alternatively, the base sets based on observation paths might make the hitting set significantly harder to solve even if the number of additional base sets is relatively small.

While we see a similar peak in the run time of most instances, the peak is significantly higher in some instances than in others. We see the highest and most pronounced peaks for two instances, case2869pegase and case3120sp. For larger l, solving these instances takes around $150 - 300$ms for larger l, where larger instances such as case6470rte generally take about $100 - 150$ms. However, the peak time for these two instances increases to almost 10 minutes, with case3120sp even exceeding the time limit of 10 minutes in one test run. In comparison, larger instances have a peak of around 1 second. To better understand this, we measured the time taken by different parts of the solver. We found that for larger l, solving hitting sets took about $20\% - 30\%$ of time for both the extreme cases and the larger cases. For the regular large cases, that share increases to around $40\% - 50\%$ for smaller l. But for the two outlier cases, solving hitting sets took over 95% and even up to 99% of total time for the longest test runs. We compared these results to the results of Bläsius et al.[BG25] for the implicit hitting set algorithm on regular PDS. Although they also report a lower performance of their solver on the same instances, the difference there is not nearly as big as the difference in our solver. Apparently, hitting sets are significantly harder to solve for these instances compared to other instances, and the difference becomes even stronger when using base sets based on observation paths in addition to only forts.

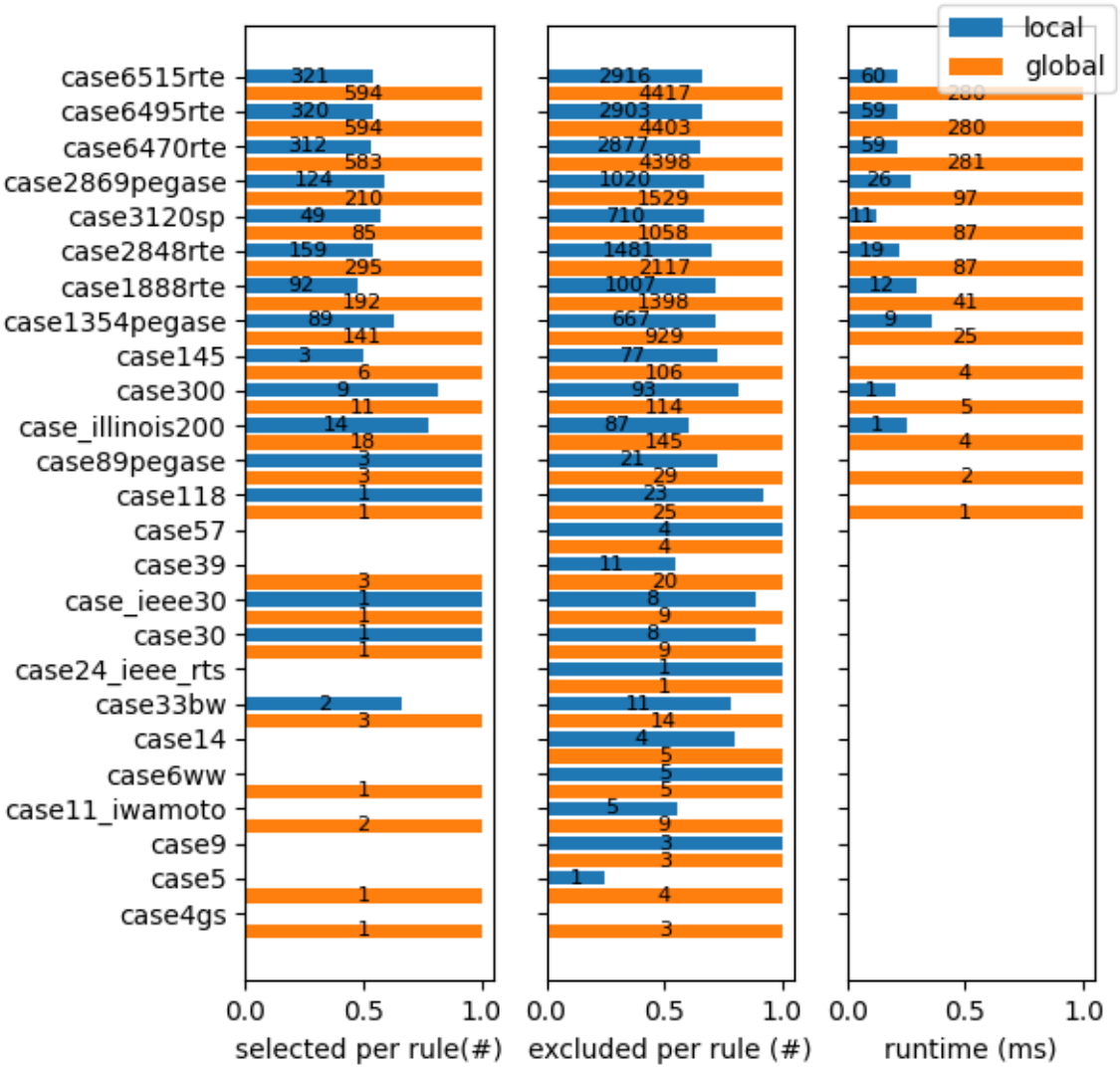


Figure 4.7.: Time taken and vertices selected or disabled by using only local reduction rules or both local and global reduction rules for $l = 8$. Both bars are normalized to the value for all reduction rules. The instances are ordered by number of vertices.

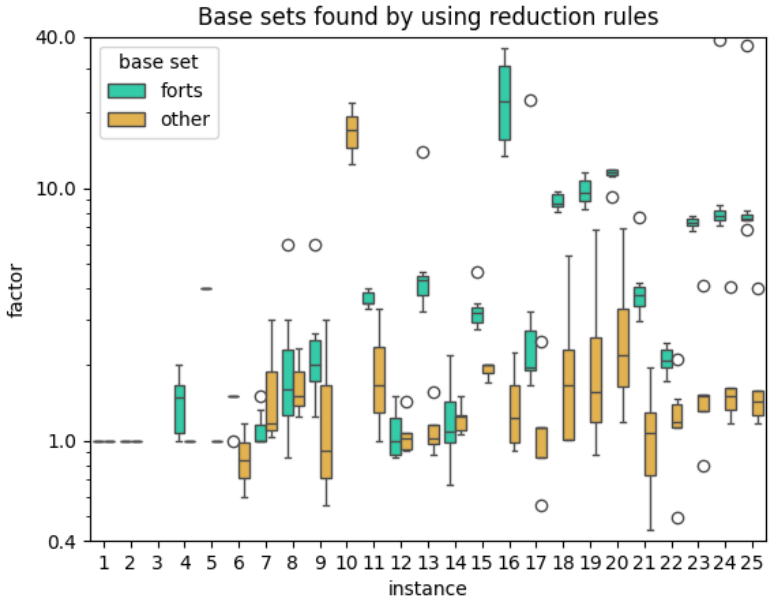


Figure 4.8.: Reduction in the number of base sets found by our solver by itself compared to number of base sets found by our solver in combination with reduction rules, aggregated over different l. *other* refers to base sets obtained over observation paths. The instances are sorted by number of vertices.

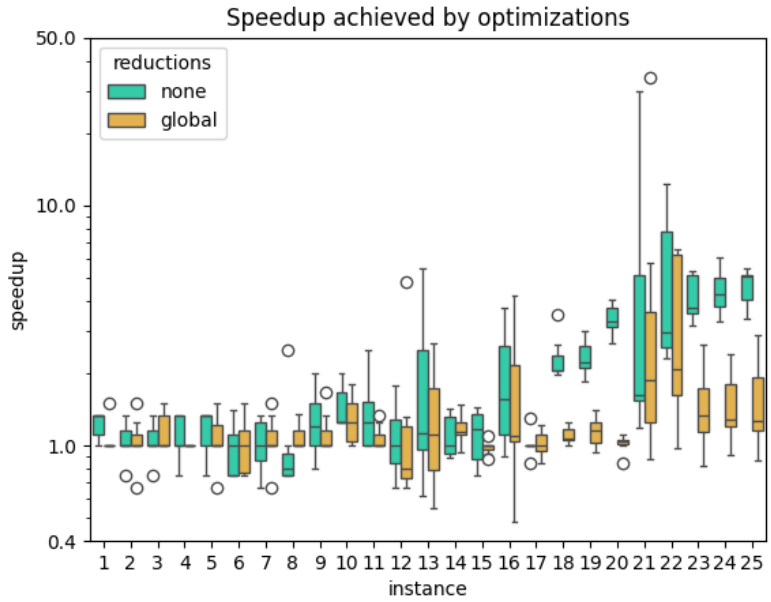
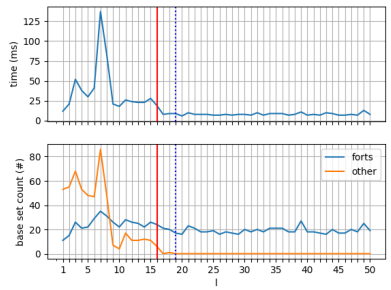
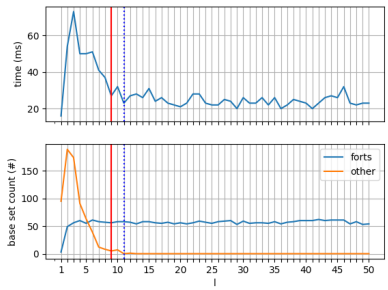


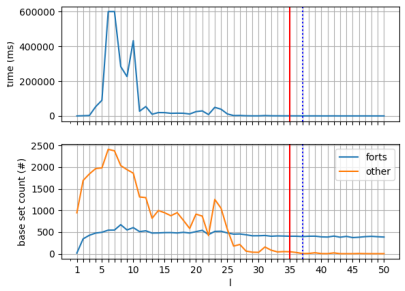
Figure 4.9.: Speed-up in the run-rime of our solver achieved by optimizing different aspects of the solver. The speed-up is aggregated over different l for each instance. For none, we used the solver without any reduction rules. For global, we used the solver with all reduction rules.



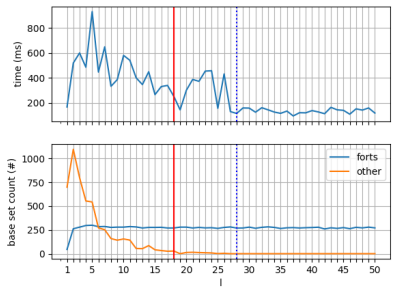
(a) Performance for case118



(b) Performance for case1354pegase

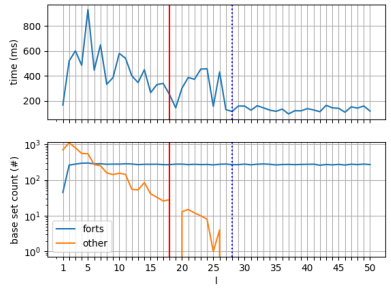


(c) Performance for case3120sp

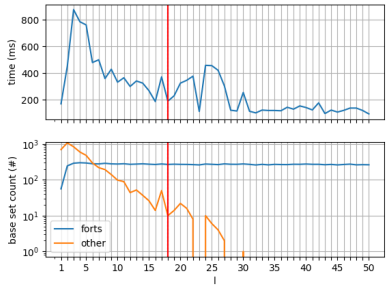


(d) Performance for case6470rte.

Figure 4.10.: Performance of our solver by l compared to the number of forts and other base sets required to find a solution for different instances. The red line marks the smallest l for which there is a minimal Power Dominating Set that can observe the graph in l rounds. The dotted blue line marks the number of rounds required to observe the entire graph for the first Power Dominating Set that our solver found without restrictions on l.



(a) Performance for case6470rte



(b) Performance for case6515rte

Figure 4.11.: Performance of our solver by l compared to the number of forts and other base sets required to find a solution for some of our largest instances. The number of base sets is given on a log scale. The red line marks the smallest l for which there is a minimal Power Dominating Set that can observe the graph in l rounds. The dotted blue line marks the number of rounds required to observe the entire graph for the first Power Dominating Set that our solver found without restrictions on l.

5. Conclusion

We show that l-round PDS is $W[2l]$ complete and thus solve its parametrized complexity. We use l-round EPDS as an intermediate problem to simulate monotone normalized circuits.

Secondly, we provide reduction rules that can be used as a pre-processing step for different solvers. We both adapt existing rules for regular PDS to work with l-round PDS as well as provide new rules that are unique for l-round PDS. These rules simplify the graph by either removing vertices or edges or limiting the choice in vertex selection.

Thirdly, we provide a solver for l-round Power Dominating Set that is based on the Implicit Hitting Set Approach. This approach has been studied for regular PDS but has never been applied to l-round PDS. Our solver generally works by finding a regular Power Dominating Set first and then refining it into a l-round Power Dominating Set for an arbitrary l. As a core concept to both our reduction rules and our solver, we introduce observation paths and potential observation paths to analyze the potential impact of selecting or unselecting certain vertices.

We ran our solver on established test instances and compared it to some existing state-of-the-art solvers. We found that our solver significantly outperforms these existing solutions, especially for larger l. We also tested our reduction rules and found that they improve the performance of our solver as well as the performance of existing solvers.

However, not all reduction rules proved to be helpful and the speed-up achieved by using the reduction rules with our solver does not come close to the speed-up reported for similar reduction rules for regular PDS. Moreover, we found that our reduction rules mainly support our solver while it is searching for a regular PDS and provide little help while it finds the actual l-round PDS. This raises the question whether it is possible to come up with reduction rules that work better with l-round PDS or whether it is just fundamentally more difficult to simplify instances for l-round PDS.

The second problem with our solver is that it performs significantly worse for small l. While it still outperforms previous solutions, the speed-up is much smaller. This result is not entirely surprising as our solver first searches for a solution to regular PDS for any l before finding a solution to l-round PDS. However, it raises the question whether a different approach to this problem could provide better results specifically for small l.

The third thing we noticed is that our solver performed significantly worse on certain instances, even compared to larger instances. This effect has also been reported for the implicit hitting set method for regular PDS on the same instances, although not to the same extent. It also occurs in different solvers for l-round PDS, although it is unclear to what extent. This raises the question of what properties of these graphs make them so difficult for our approach.

Bibliography

[25] *Gurobi Optimizer Reference Manual*. 11th ed. 2025.

[Aaz10] Ashkan Aazami. "Domination in graphs with bounded propagation: algorithms, formulations and hardness results". In: *J Comb Optim, 19* (2010), pp. 429–456. DOI: *10.1007/s10878-008-9176-7*.

[BFH18] Boris Brimkov, Caleb C. Fast, and Illya V. Hicks. *Computational Approaches for Zero Forcing and Related Problems*. 2018. arXiv: *1704.02065*.

[BG25] Thomas Bläsius and Max Göttlicher. "An Efficient Algorithm for Power Dominating Set". In: *Algorithmica, 87* (2025), pp. 344–376. DOI: *10.1007/s00453-024-01283-8*.

[BMBA93] T.L. Baldwin, L. Mili, M.B. Boisen, and R. Adapa. "Power system observability with minimal phasor measurement placement". In: *IEEE Transactions on Power Systems* Volume 8 (1993), pp. 707–715. DOI: *10.1109/59.260810*.

[BMS19] Boris Brimkov, Derek Mikesell, and Logan Smith. "Connected power domination in graphs". In: *Journal of Combinatorial Optimization* Volume 38 (2019), pp. 292–315. DOI: *10.1007/s10878-019-00380-7*.

[Bru93] Dennis J. Brueni. "Minimal PMU placement for graph observability: a decomposition approach". In: (1993).

[DFR98] Rodney G. Downey, Michael R. Fellows, and Kenneth W. Regan. "Parameterized circuit complexity and the W hierarchy". In: *Theoretical Computer Science* Volume 191 (1998), pp. 97–115. ISSN: 0304-3975. DOI: *https://doi.org/10.1016/S0304-3975(96)00317-9*.

[JV20] Raka Jovanovic and Stefan Voss. "The fixed set search applied to the power dominating set problem". In: *Expert Systems* Volume 37 (2020), e12559. eprint: *https://onlinelibrary.wiley.com/doi/pdf/10.1111/exsy.12559*.

[KTX17] Iyad Kanj, Dimitrios M. Thilikos, and Ge Xia. "On the parameterized complexity of monotone and antimonotone weighted circuit satisfiability". In: *Information and Computation* Volume 257 (2017), pp. 139–156. ISSN: 0890-5401. DOI: *https://doi.org/10.1016/j.ic.2017.11.002*.

[Lia16] Chung-Shou Liao. "Power domination with bounded time constraints". In: *Journal of Combinatorial Optimization* Volume 31 (2016), pp. 725–742. DOI: *10.1007/s10878-014-9785-2*.

[SH20] Logan A. Smith and Illya V. Hicks. *Optimal Sensor Placement in Power Grids: Power Domination, Set Covering, and the Neighborhoods of Zero Forcing Forts*. 2020. arXiv: *2006.03460*.

[Thu+18] Leon Thurner, Alexander Scheidler, Florian Schäfer, Jan-Hendrik Menke, Julian Dollichon, Friederike Meier, Steffen Meinecke, and Martin Braun. "Pandapower—An Open-Source Python Tool for Convenient Modeling, Analysis, and Optimization of Electric Power Systems". In: *IEEE Transactions on Power Systems* Volume 33 (2018), pp. 6510–6521. DOI: *10.1109/TPWRS.2018.2829021*.

A. Appendix

A.1. Integer Linear Programming formulations

A.1.1. Model 1 (Jovanovic et al.)

We adapt the MILP formulation provided by Jovanovic et al.[JV20] in the variation by Bläsius et al. [BG25] to work for l-round EPDS. That formulation represents each vertex v with two variables x_v and s_v. The first variable indicates whether a vertex is selected, and the second variable counts the number of propagation steps required to observe a vertex. Since that formulation already counts the number of propagation steps, which is essentially the same as counting the number of rounds of parallel propagation, we only need to add a constraint to limit the number of steps. We provide the full formulation. Note that we only changed rule (6).

$$
\begin{aligned}
(IP_l) \qquad & min \sum_v x_v & \\
\text{s.t.} & & \\
(1) \qquad & s_v \leq x_t + M(1 - x_t) & \forall v \in V, \forall t \in N[t] \\
(2) \qquad & s_v \leq M(x_v + \sum_{w \in N(v)} (x_w + p_{w,v})) & \forall v \in V \\
(3) \qquad & \sum_{w \in N(v)} p_{w,v} \leq 1 & \forall v \in V \\
(4) \qquad & \sum_{w \in N(v)} p_{v,w} \leq 1 & \forall v \in V \\
(5) \qquad & s_v \geq s_t + 1 - M(1 - p_{w,v}) & \forall v \in V, w \in N(v), t \in N[w] \backslash \{v\} \\
(6) \qquad & 1 \leq s_v \leq L & \forall v \in V \\
(7) \qquad & x_v = 1 & \forall v \in X \\
(8) \qquad & x_v = 0 & \forall v \in V \backslash Y \\
(9) \qquad & p_{v,w} = 0 & \forall v \in V \backslash W, w \in N(v) \\
(10) \qquad & x_v, p_{w,v} \in \{0, 1\} &
\end{aligned}
$$

A.1.2. Model 2 (Aazami, corrected)

We adapt the MILP formulation provided by Aazami et al. [Aaz10] to work for an extension variant of l-round PDS. That formulation represents each vertex v with a variable x_v and l different variables z_v^i. The variable x_v indicates whether v is selected and z_v^i indicates whether v is observed in round i. Additionally, there are l variables $Y_{u \to v}^i$ for every pair (u, v) of adjacent vertices that indicate whether u can propagate to v in round i. However, this

formulation can be incorrect in some cases. It includes a rule that forces at least one new vertex to be observed every round, so it does not accept solutions that take less than l rounds of propagation to observe the graph. We modify rule (9) to also accept solutions that take less than l rounds of propagation. The modified rule still forces additional vertices to be observed as long as not all vertices are observed, but is also satisfied if all vertices are observed.

Furthermore, we adapt the formulation to work for instances of the extension variant of l-round PDS. For this, we add constraints that ensure that pre-selected vertices are always selected and excluded vertices are never selected. We also add constraints that propagation is limited to propagating vertices. As with our reduction rules and solver, we limit this formulation to instances with no pre-observed vertices and only target vertices. The rules (4), (6) and (7) are added for the extension variant. We provide the full formulation here.

Let (G, W, X, Y) be an instance of l-round EPDS with graph $G = (V, E)$, pre-selected vertices X, selectable vertices Y and propagating vertices W. Let $\delta(G)$ be the size of the smallest closed neighborhood in G and $T = \{1, ..., l\}$ the set of valid rounds of propagation.

$$(IP_l) \qquad min \sum_v x_v$$

s.t.

$$(1) \qquad 1 \leq z_v^l \qquad \forall v \in V$$

$$(2) \qquad z_v^1 \leq \sum_{u \in N[v]} x_u \qquad \forall v \in V$$

$$(3) \qquad Y_{u \to v}^t \leq z_w^t \qquad \forall (u,v) : \{u,v\} \in E, \forall w \in N[u] \backslash \{v\}, \forall t \in T$$

$$(4) \qquad Y_{u \to v}^t \leq 0 \qquad \forall (u,v) : \{u,v\} \in E, \forall u \in V \backslash W, \forall t \in T$$

$$(5) \qquad z_v^t \leq \sum_{u \in N(v)} Y_{u \to v}^{t-1} + x_v \qquad \forall v \in V, \forall t \in T \backslash \{1\}$$

$$(6) \qquad x_v \leq 0 \qquad \forall v \in V \backslash Y$$

$$(7) \qquad 1 \leq x_v \qquad \forall v \in X$$

$$(8) \qquad \delta(G) + 1 \leq \sum_{v \in V} z_v^1$$

$$(9) \qquad N \leq \sum_{v \in V} z_v^t + N \sum_{v \in V} (z_v^t - z_v^{t-1}) \qquad \forall t \in T \backslash \{1\}$$

$$(10) \qquad \text{all variables are binary}$$

A.1.3. Model 3 (Brimkov et al.)

We adapt the MILP formulation by Brimkov et al.[BMS19] to work for an extension variant of l-round PDS. This model is designed for a directed graph. For an instance of l-round PDS with an undirected graph, it considers the directed graph that is obtained by replacing every undirected edge $\{u, v\}$ with two directed edges (u, v) and (v, u).

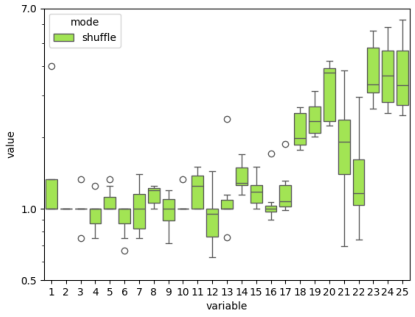
The model represents each vertex v with two variables s_v and x_v. The variable s_v indicates whether v is selected, and the variable x_v indicates the round in which v becomes observed. If v is selected, that round is actually defined as 0 instead of 1. Every directed edge $e = (u, v)$ is represented by a variable y_e that indicates whether the vertex v is observed by domination or

propagation from u. We add additional constraints (6) and (7) to enforce that pre-selected vertices are always and excluded vertices are never selected. We add rule (8) to ensure that non-propagating vertices can only observe other vertices with the domination rule.

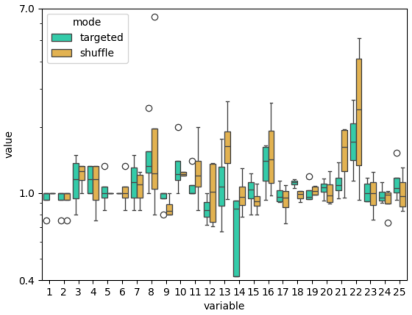
Let (G, W, X, Y) be an instance of l-round EPDS with directed edges E, pre-selected vertices X, selectable vertices Y and propagating vertices W.

$$
\begin{aligned}
(IP_l) \qquad & min \sum_v s_v & & \\
\text{s.t.} & & & \\
(1) \qquad & s_v + \sum_{e \in \delta^-(v)} y_e = 1 & & \forall v \in V \\
(2) \qquad & x_u - x_v + (L+1) \cdot y_e \leq L & & \forall e = (u,v) \in E \\
(3) \qquad & x_w - x_v + (L+1) \cdot y_e \leq L + (L+1) \cdot s_u & & \forall e = (u,v) \in E, \forall w \in N(u) \backslash \{v\} \\
(4) \qquad & s_v = 1 & & \forall v \in X \\
(5) \qquad & s_v = 0 & & \forall v \in V \backslash Y \\
(6) \qquad & y_e \leq s_u & & \forall e = (u,v) \in E
\end{aligned}
$$

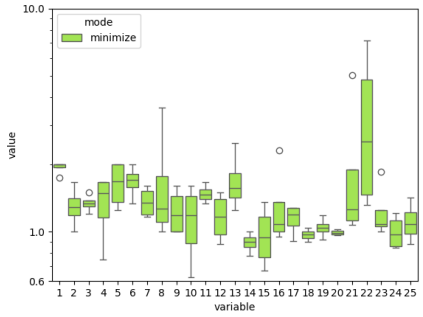
A.2. Experiments



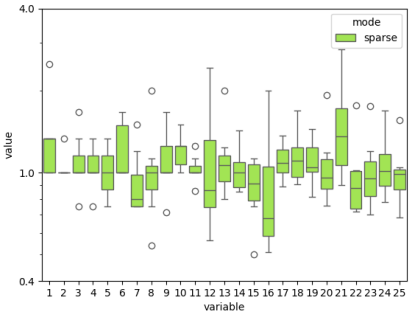
(a) Speed-up achieved by using a more sophisticated method for generating fort neighborhoods as detailed in Section 4.3.1.



(b) Speed-up achieved by using more sophisticated methods for generating base sets based on observation paths as detailed in Section 4.3.1. *Targeted* refers to the simpler method of choosing certain vertices to find base sets, *shuffle* refers to the method that is similar to that used for finding fort neighborhood. We only measured for $l \leq 8$ since the solver generally finds very few base sets based on observation paths for larger l.



(c) Speed-up achieved by only adding inclusion minimal base sets based on observation paths. We only measured for $l \leq 8$ since the solver generally finds very few base sets based on observation paths for larger l.



(d) Speed-up achieved by adding the l-neighborhood of some vertices as base sets.

Figure A.1.: Speed-up achieved by using various optimizations for our solver. The solver is always used by itself without any reduction rules. The speed-up is aggregated over runs for different l for every instance.