Distributed algorithms for edge dominating sets

Suomela, Jukka

ACM
2010


http://hdl.handle.net/10138/27921
https://doi.org/10.1145/1835698.1835783

Downloaded from Helda, University of Helsinki institutional repository.

This is an electronic reprint of the original article.

This reprint may differ from the original in pagination and typographic detail.

Please cite the original version.
Distributed Algorithms for Edge Dominating Sets

Jukka Suomela
Helsinki Institute for Information Technology HIIT, University of Helsinki
P.O. Box 68, FI-00014 University of Helsinki, Finland
jukka.suomela@cs.helsinki.fi

ABSTRACT
An edge dominating set for a graph \( G \) is a set \( D \) of edges such that each edge of \( G \) is in \( D \) or adjacent to at least one edge in \( D \). This work studies deterministic distributed approximation algorithms for finding minimum-size edge dominating sets. The focus is on anonymous port-numbered networks: there are no unique identifiers, but a node of degree \( d \) can refer to its neighbours by integers 1, 2, ..., \( d \). The present work shows that in the port-numbering model, edge dominating sets can be approximated as follows: in \( d \)-regular graphs, to within \( 4 - \frac{1}{d+1} \) for an odd \( d \) and to within \( 4 - \frac{2}{\Delta} \) for an even \( \Delta \). These approximation ratios are tight for all values of \( d \) and \( \Delta \): there are matching lower bounds.

Categories and Subject Descriptors
C.2.4 [Computer-Communication Networks]: Distributed Systems; F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems—computations on discrete structures

General Terms
Algorithms, Theory

1. INTRODUCTION
This work studies the approximability of the edge dominating set problem from the perspective of deterministic (non-randomised) distributed algorithms in anonymous port-numbered networks.

1.1 Edge Dominating Sets and Matchings
Let \( G \) be a simple, undirected graph with the edge set \( E_G \). A set \( D \subseteq E_G \) of edges is an edge dominating set for \( G \) if each edge \( e \in E_G \setminus D \) is adjacent to at least one edge in \( D \). See Figure 1 for examples.

By definition, any maximal matching is an edge dominating set. An edge dominating set is not necessarily a matching; however, given an edge dominating set \( D \), it is straightforward to construct a maximal matching with at most \( |D| \) edges [25]. Hence a minimum maximal matching (a maximal matching with the smallest possible number of edges) is also a minimum edge dominating set.

This is a corollary of a more general result due to Allan and Laskar [1]: if a graph is claw-free (no induced subgraph \( K_{1,3} \)), then a minimum maximal independent set is also a minimum dominating set. The line graph \( L(G) \) of any graph \( G \) is claw-free, the dominating sets of \( L(G) \) correspond to the edge dominating sets of \( G \), and the maximal independent sets of \( L(G) \) correspond to the maximal matchings of \( G \).

1.2 Centralised Polynomial-Time Algorithms
From the perspective of centralised polynomial-time algorithms, the problem of finding a minimum edge dominating set is equivalent to the problem of finding a minimum maximal matching. Both of these are NP-hard optimisation problems [25], and they are hard to approximate to within factor \( 7/6 - \epsilon \) [9]. The problem of finding a minimum-weight edge cover is as hard to approximate as minimum-weight vertex cover [8].

The connection between matchings and edge dominating sets implies a simple 2-approximation algorithm: any maximal matching is a 2-approximation of a minimum edge dominating set. Approximating minimum-weight edge dominating sets is less straightforward, but Fujito and Nagamochi [12] show how to find a 2-approximation. Polynomial-time approximation schemes are known for planar graphs [6] and civilised graphs [15].

1.3 Distributed Algorithms
Edge dominating sets have received little attention in the distributed computing community. However, some results related to matchings and independent sets have straightfor-
ward corollaries that concern the distributed approximability of edge dominating sets.

On the positive side, one can again take any algorithm that finds a maximal matching and apply it to find a 2-approximation of a minimum edge dominating set. For example, if we have unique node identifiers in the network, we can use the deterministic algorithms by Hančkowiak et al. [14] and Panconesi and Rizzi [19], with running times $O(\log n)$ and $O(\Delta + \log n)$ communication rounds, respectively; here $n$ is the number of nodes and $\Delta$ is the maximum degree.

The running times of these algorithms depend on $n$, and this is unavoidable if we want to achieve an approximation factor better than 3. Czygrinow et al. [10] and Lenzen and Wattenhofer [17] show that finding a constant-factor approximation of a maximum independent set in a cycle requires $\Omega(\log n)$ communication rounds, and a simple local reduction [22] gives the same lower bound for finding a factor $3 - \epsilon$ approximation of a minimum edge dominating set.

### 1.4 Algorithms in Port-Numbered Networks

The above results deal with deterministic distributed algorithms in networks with unique node identifiers. This work studies a strictly weaker model of computation: deterministic distributed algorithms in anonymous networks in the port-numbering model: there are no node identifiers, but a node of degree $d$ can refer to its neighbours by integers $1, 2, \ldots, d$. See Section 2 for a formal definition of the model.

Computation in synchronous port-numbered networks has been studied for decades; one of the pioneers was Angluin [2] in 1980. However, the main focus has been on global problems such as leader election [2, 24], construction of spanning trees [24], computation of functions that depend on all nodes [5, 7, 23], recognition of topological properties [2, 24], and graph exploration and rendezvous [16]. Such problems typically require $\Omega(n)$ communication rounds – or, in many cases, are unsolvable in the port-numbering model.

Much less is known about graph problems that are of a more local nature and have potential for efficient, highly scalable distributed algorithms. Classical packing problems such as matchings and independent sets are typically unsolvable for trivial reasons, but covering problems are more promising. Node-based covering problems (the task is to choose a subset of nodes that “covers” the graph) have been studied in prior work: for example, the vertex cover problem can be approximated within factor 2 in the port-numbering model in bounded-degree graphs [3, 4], and this approximation guarantee is tight. However, it seems that edge-based covering problems have not been studied previously in this model.

### 1.5 Contributions

The contributions are summarised in Table 1. This work presents a complete characterisation of the deterministic approximability of edge dominating sets in the port-numbering model, both in graphs of maximum degree $\Delta$ and in $d$-regular graphs, for all values of the parameters $\Delta$ and $d$. All approximation ratios are tight: there are exactly matching upper and lower bounds.

On a more conceptual level, the contributions are twofold. First, this work highlights the different nature of edge-based covering problems, in comparison with node-based covering problems. Informally, in a regular port-numbered graph, all nodes may look identical from the perspective of a distributed algorithm, but all edges do not look identical to each other. Tight lower bound constructions for covering problems such as vertex covers and dominating sets are typically trivial: a cycle or a complete graph will do. This is not the case with edge-based problems.

Second, this work gives yet another example of the close connection between the port-numbering model and local algorithms. In a strictly local algorithm, the running time does not depend on the number of nodes [18, 22]. Even though the negative results hold for any algorithm, regardless of its running time, the matching positive results are local algorithms: the running times depend on the parameters $d$ and $\Delta$, but they are independent of $n$. Indeed, these algorithms are the best known deterministic local algorithms for the edge dominating set problem – it is not known if a better approximation ratio can be achieved in constant time with the help of unique node identifiers.

### Table 1: Approximability of edge dominating sets: the best possible approximation ratios that can be achieved by any deterministic distributed algorithm in the port-numbering model.

<table>
<thead>
<tr>
<th>Graph family</th>
<th>Approx. ratio</th>
<th>Lower bound</th>
<th>Upper bound</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d$-regular graphs:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$d = 1, 3, \ldots$</td>
<td>$4 - \frac{6}{d+1}$</td>
<td>Theorem 2</td>
<td>Theorem 4</td>
<td>$O(d^2)$</td>
</tr>
<tr>
<td>$d = 2, 4, \ldots$</td>
<td>$4 - \frac{2}{d}$</td>
<td>Theorem 1</td>
<td>Theorem 3</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>graphs with maximum degree $\Delta$:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Delta = 1$</td>
<td>1</td>
<td>trivial</td>
<td>trivial</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>$\Delta = 3, 5, \ldots$</td>
<td>$4 - \frac{2}{\Delta - 1}$</td>
<td>Corollary 1</td>
<td>Theorem 5</td>
<td>$O(\Delta^2)$</td>
</tr>
<tr>
<td>$\Delta = 2, 4, \ldots$</td>
<td>$4 - \frac{2}{\Delta}$</td>
<td>Corollary 1</td>
<td>Theorem 5</td>
<td>$O(\Delta^2)$</td>
</tr>
</tbody>
</table>

2. PRELIMINARIES

Let $G$ be a simple, undirected graph with the node set $V_G$ and the edge set $E_G$. An edge $e = \{u, v\} \in E_G$ is said to cover the nodes $u$ and $v$, and an edge $e_1 \in E_G$ is said to dominate any edge $e_2 \in E_G$ that is adjacent to $e_1$, including $e_1$ itself. These terms are generalised to sets of edges and nodes in a natural manner: for example, a set $C \subseteq E_G$ of edges covers a set of nodes $X \subseteq V_G$ if for each $v \in X$ there is an $e \in C$ that covers $v$. An edge cover is a set $C \subseteq E_G$ that covers $V_G$ and an edge dominating set is a set $D \subseteq E_G$ that dominates $E_G$.

A set $M \subseteq E_G$ is a matching if each node $v \in V_G$ is incident to at most one edge of $M$. More generally, a set $M \subseteq E_G$ is a $k$-matching if each node $v \in V_G$ is incident to at most $k$ edges in $M$; put otherwise, a subgraph induced by a $k$-matching is a graph of maximum degree at most $k$. In particular, the subgraph induced by a 2-matching consists of paths and cycles. A matching is maximal if it is not a proper subgraph of a matching.
A k-factor of $G$ is a k-regular spanning subgraph $H$ of $G$: we have the same node set $V_H = V_G$, and each node $v \in V_H$ has degree $k$ in $H$. For example, a 1-factor forms a perfect matching, and a 2-factor is a collection of disjoint cycles that span $V_G$.

A k-factorisation of $G$ is a collection $G(1), G(2), \ldots, G(c)$ of $k$-factors of $G$ such that each edge $e \in E_G$ is in exactly one $E_G(i)$; that is, a $k$-factorisation partitions the edge set into $k$-factors. For example, a 1-factorisation of a $d$-regular graph $G$ can be interpreted as a $d$-colouring of the edges of $G$: each factor is a colour class.

Not all graphs admit a k-factorisation; an obvious necessary condition is that the graph $G$ has to be $ck$-regular for some $c$. In the case of 1-factorisations, this condition is not sufficient: there are regular graphs that cannot be 1-factorised (e.g., an odd cycle). However, in the case of 2-factorisations, this condition turns out to be sufficient. A 120-year-old result due to Petersen [20] shows that any 2-regular graph admits a 2-factorisation – see, e.g., Diestel [11, p. 39] for a modern proof.

### 2.1 Port-Numbered Graphs

A port-numbered graph $G$ is defined by a set of nodes $V_G$ and two functions, $d_G : V_G \to \mathbb{N}$ and $p_G : V_G \to P_G$, where

$$P_G = \{(v, i) : v \in V_G, i \in P_G(v)\},$$

$$P_G = \{1, 2, \ldots, d_G(v)\}.$$

It is required that $p_G$ is an involution, i.e., a bijection that is its own inverse.

The integer $d_G(v)$ is called the degree of the node $v \in V_G$. Each $(v, i) \in P_G$ is a port. If $p_G(v, i) = (u, j)$, we say that the port $i$ of $v$ is connected to the port $j$ of $u$. Figure 2a shows two examples of port-numbered graphs.

Given the involution $p_G$, we can define the multiset of edges $E_G$ as follows: For each pair of ports $(v, i), (u, j) \in P_G$ with $p_G(v, i) = (u, j)$ and $(v, i) \neq (u, j)$, we have an undirected edge $(v, u) \in E_G$, and for each fixed point $(v, i) \in

![Figure 2: Examples of port-numbered graphs: a simple graph $H$ and a multigraph $M$. For example, $V_M = \{s, t\}$, $d_M(s) = 3$, $d_M(t) = 4$, and the involution $p_M$ maps $(s, 1) \leftrightarrow (t, 2), (s, 2) \leftrightarrow (t, 1), (s, 3) \leftrightarrow (s, 3)$, and $(t, 3) \leftrightarrow (t, 4)$. (a) Ports (boxes) and connections (arrows). (b) Nodes (circles), undirected edges (lines), and directed edges (arrows).](image)

2.2 Model of Computation

In a synchronous distributed algorithm, computation proceeds in synchronous communication rounds. In each round, the following operations are performed in a port-numbered graph $G$: (i) each node performs local computation, (ii) each node $v \in V_G$ sends one message to each port $i \in P_G(v)$, and (iii) each node $v \in V$ receives one message from each port $i \in P_G(v)$. The involution $p_G$ indicates how the messages are routed: if $p_G(v, i) = (u, j)$, then the message sent by $v$ to its port $i$ is received by $u$ from its port $j$.

All nodes run the same deterministic distributed algorithm $A$. Initially, each node $v \in V_G$ knows only its degree $d_G(v)$. After each round, a node may decide to stop computation and announce its output. The running time of the algorithm $A$ is the maximum number of synchronous rounds until all nodes have stopped.

When we use a distributed algorithm $A$ to find an edge dominating set $D$ in a simple port-numbered graph $G$, we assume that each node $v \in V_G$ outputs a subset $X(v) \subseteq P_G(v)$ of port numbers; if $i \in X(v)$ and $p_G(v, i) = (u, j)$ then the edge $(u, v)$ is in the set $D$. Naturally, we require that the output is internally consistent: if $i \in X(v)$ and $p_G(v, i) = (u, j)$, then we must also have $j \in X(u)$.

### 2.3 Covering Maps

Let $G$ and $H$ be two port-numbered graphs. A surjection $f : V_H \to V_G$ is a covering map from $H$ to $G$ if (i) it preserves the degrees, i.e., $d_H(v) = d_G(f(v))$ for all $v \in V_H$, and (ii) it preserves the connections, i.e., $p_H(v, i) = (u, j)$ implies $p_G(f(v), i) = (f(u), j)$ for all $(v, i) \in P_G$. If there exists a covering map from $H$ to $G$, then $H$ is a covering graph of $G$. See Figure 3 for an example.

A key observation is that if we apply any deterministic distributed algorithm $A$ both in the port-numbered graph $G$ and in its covering graph $H$, then the output of a node $v \in V_H$ is necessarily identical to the output of the node $P_G$ with $p_G(v, i) = (v, i)$, we have a directed loop $(v, v) \in E_G$; see Figure 2b for an illustration.

This way we can interpret any port-numbered graph $G$ as a graph with the node set $V_G$ and the edge set $E_G$, and we can also apply the usual graph-theoretic terminology; for example, a port-numbered graph $G$ is simple if the edge set $E_G$ does not contain loops or multiple parallel edges. Conversely, we can take any undirected graph $G$ with the node set $V_G$ and the edge set $E_G$, and turn $G$ into a port-numbered graph by constructing an involution $p_G$ that is compatible with $E_G$.

![Figure 3: An example of a covering graph. The simple port-numbered graph $C$ is a covering graph of the multigraph $M$. The covering map $f$ maps each grey node of $C$ to the grey node of $M$, and each white node of $C$ to the white node of $M$.](image)
f(v) ∈ V_G. To see this, note that the initial state of a node v ∈ V_H is identical to the initial state of the node f(v) ∈ V_G, as both of them run the same algorithm A. Now assume inductively that before the communication round t, for each v ∈ V_H the local state of v in H is the same as the local state of f(v) in G. Then during the round t, for each (v, i) ∈ P_H the message sent to the port (v, i) in H equals the message sent to (f(v), i) in G. Since the covering map preserves the connections, it follows that for each (v, i) ∈ P_H the message received from the port (v, i) in H equals the message received from (f(v), i) in G; hence after the round t, the local state of v ∈ V_H is identical to the local state of f(v) ∈ V_G. Whenever the node v decides to stop and announce its output, the node f(v) also stops and produces the same output.

3. LOWER BOUND CONSTRUCTION: EVEN DEGREE

In this section we prove the following theorem.

**Theorem 1.** For each d = 2, 4, . . . , there is a d-regular port-numbered graph G such that no deterministic distributed algorithm can achieve a better approximation ratio than 4 − 2/d for the minimum edge dominating set problem in G.

3.1 Graph

The graph G is constructed as follows (see Figure 4 for an illustration in the case d = 6). The node set is V_G = A ∪ B where

A = \{a_1, a_2, \ldots, a_d\}, \quad B = \{b_1, b_2, \ldots, b_{d−1}\}.

The edge set is E_G = S ∪ T where

S = \{(a_1, a_2), (a_3, a_4), \ldots, (a_{d−1}, a_d)\},

T = \{(a_i, b_j) : a_i ∈ A, b_j ∈ B\}.

The graph G is d-regular. The subgraph induced by S is a matching and the subgraph induced by T is the complete bipartite graph K_{d,d−1}. By construction, S is an edge dominating set: each edge in T is adjacent to an edge in S. Moreover, S is an optimal edge dominating set, since |E_G| = (2d − 1)|S| and each edge can dominate at most 2d − 1 edges.

3.2 Port Numbering

Let k = d/2. Since G is 2k-regular, we can 2-factorise it; let the factors be G(1), G(2), . . . , G(k) – see Figure 4 for an example. Each subgraph G(i) consists of cycles. Let H(i) be an orientation of G(i) that consists of directed cycles – that is, for each (u, v) ∈ E_G(i) there is either (u, v) or (v, u) in E_H(i), and the outdegree and indegree of each node v of H(i) is 1.

Now we are ready to define a port numbering p_G for G. For each i = 1, 2, . . . , k and for each (u, v) ∈ E_H(i), we set p_G(u, 2i − 1) = (v, 2i) and conversely p_G(v, 2i) = (u, 2i − 1); see Figure 4.

3.3 Covering Map

Let M be a port-numbered multigraph with one node V_M = \{x\} of degree d_M(x) = 2k. The involution p_M maps (x, 2i − 1) ↔ (x, 2i) for each i = 1, 2, . . . , k. See Figure 4 for an illustration in the case d = 6. Define the function f : V_G → V_M by setting f(v) = x for all v ∈ V_G; it can be checked that f is a covering map from G to M.

3.4 Approximation Ratio

Let A be a deterministic distributed algorithm that finds an edge dominating set in any port-numbered 2k-regular graph, and apply A to G; let D be the edge dominating set produced by A. Since D ≠ ∅, there is an edge e ∈ D; let G(i) be the 2-factor with e ∈ E_G(i). Hence there is a node a that outputs a set X(a) which contains the port number 2i − 1 and another node b that outputs a set X(b) which contains the port number 2i.

The covering map f shows that all nodes of G produce the same output. Hence all nodes v ∈ V_G output a set X(v) with |2i − 1, 2i| ⊆ X(v), and the dominating set D has to contain all edges of the factor G(i). We conclude that the approximation ratio of A is at least

\[ \frac{|D|}{|S|} \geq \frac{|E_G(i)|}{|S|} = \frac{|V_G|}{|S|} = \frac{4k - 1}{k}, \]

which completes the proof of Theorem 1.
The edge set is $E$. The subgraph induced by $-4$. Figure 5: The component $H(\ell)$ used in the proof of Theorem 2 for $d = 5$. The thick lines indicate the set $S(\ell)$.

4. LOWER BOUND CONSTRUCTION: ODD DEGREE

In this section we prove the following theorem.

**Theorem 2.** For each $d = 1, 3, \ldots$, there is a $d$-regular port-numbered graph $G$ such that no deterministic distributed algorithm can achieve a better approximation ratio than $k$.

4.1 Graph

Let $k = (d - 1)/2$. For each $\ell = 1, 2, \ldots, d$, we first construct a $2k$-regular graph $H(\ell)$ as follows (see Figure 5 for an illustration in the case $d = 5$). The node set is $V_{H(\ell)} = A(\ell) \cup B(\ell) \cup C(\ell)$ where

$$A(\ell) = \{a_{\ell,1}, a_{\ell,2}, \ldots, a_{\ell,2k}\}, \quad C(\ell) = \{c_\ell\},$$

$$B(\ell) = \{b_{\ell,1}, b_{\ell,2}, \ldots, b_{\ell,2k}\}.$$

The edge set is $E_{H(\ell)} = R(\ell) \cup S(\ell) \cup T(\ell)$ where

$$R(\ell) = \{c_\ell, b_{\ell,i} : b_{\ell,i} \in B(\ell)\},$$

$$S(\ell) = \{a_{\ell,1}, a_{\ell,2}\}, \{a_{\ell,3}, a_{\ell,4}\}, \ldots, \{a_{\ell,2k-1}, a_{\ell,2k}\},$$

$$T(\ell) = \{a_{\ell,i}, b_{\ell,j} : a_{\ell,i} \in A(\ell), b_{\ell,j} \in B(\ell), i \neq j\}.$$

The subgraph induced by $R(\ell)$ is a star, the subgraph induced by $S(\ell)$ is a matching, and the subgraph induced by $T(\ell)$ is a crown graph (a complete bipartite graph minus a perfect matching).

Since the graph $H(\ell)$ is $2k$-regular, we can again find a $2$-factorisation and hence construct a port numbering $p_{H(\ell)}$ so that for each node $u$ and each $i = 1, 2, \ldots, 2k$, the port $2i - 1$ of $u$ is connected to the port $2i$ of an adjacent node $v$ and vice versa; see Figure 5 for an example.

The port-numbered graph $G$ contains the port-numbered components $H(\ell)$ for each $\ell = 1, 2, \ldots, d$ as subgraphs. The node set of $G$ consists of the node sets of the components $H(\ell)$ and the sets

$$P = \{p_1, p_2, \ldots, p_d\}, \quad Q = \{q_1, q_2, \ldots, q_{2k}\}.$$

We create the following connections, in addition to those inherited from the components $H(\ell)$:

$$(p_\ell, \ell) \leftrightarrow (c_\ell, d) \quad \forall \ell = 1, 2, \ldots, d,$$

$$(p_\ell, \ell) \leftrightarrow (b_{\ell,i}, d) \quad \forall \ell = 1, 2, \ldots, d, \quad i = 1, 2, \ldots, 2k, \; i \neq \ell,$$

$$(p_\ell, \ell) \leftrightarrow (b_{\ell,i}, d) \quad \forall \ell = 1, 2, \ldots, d,$$

$$(q_\ell, \ell) \leftrightarrow (a_{\ell,i}, d) \quad \forall \ell = 1, 2, \ldots, d, \quad i = 1, 2, \ldots, 2k.$$

See Figure 6 for an illustration in the case $d = 5$. Note, in particular, that each edge that joins a node $u \in P \cup Q$ to a node $v \in V_{H(\ell)}$ connects the port $\ell$ of $u$ to the port $d$ of $v$.

Figure 6: The graph $G$ constructed in the proof of Theorem 2 for $d = 5$. The thick lines indicate the optimal dominating set $D^*$. For clarity, only a subset of edge is shown: edges of each $H(\ell)$, edges connected to $H(1)$, edges connected to $p_1$, and edges in $Y$. The components $H(\ell)$ are illustrated in Figure 5 in more detail.
4.2 Optimal Solution

Define the edge sets
\[ Y = \{ \{v, u\} : \ell = 1, 2, \ldots, d \}, \quad D' = Y \cup \bigcup_{\ell} S(\ell). \]
Now \( D' \) is an optimal edge dominating set for the graph \( G \); each edge \( e \notin D' \) is adjacent to exactly one edge in \( D' \). By construction, \( |D'| = (k+1)d \).

4.3 Covering Map

Let \( M \) be a port-numbered multigraph with the node set
\[ V_M = \{ x_1, x_2, \ldots, x_d, y \}. \]
All nodes \( v \in V_M \) have degree \( p_M(v) = d \). The involution \( p_M \) maps
\[
(x_\ell, 2\ell - 1) \leftrightarrow (x_\ell, 2\ell) \quad (y, \ell) \leftrightarrow (x_\ell, d) \quad \forall \ell = 1, 2, \ldots, d, \]
See Figure 7 for an illustration in the case \( d = 5 \).
Define the function \( f : V_G \rightarrow V_M \) as follows:
\[
f(v) = x_\ell \quad \text{for each } \ell = 1, 2, \ldots, d, \quad v \in V_{H(\ell)},
f(v) = y \quad \text{for each } v \in P \cup Q.
\]
It can be checked that \( f \) is a covering map from \( G \) to \( M \). Hence we have partitioned the node set of \( G \) in \( d+1 \) equivalence classes and the edge set of \( G \) in \((k+1)d \) equivalence classes.

4.4 Approximation Ratio

Assume that \( A \) is a deterministic distributed algorithm that finds an edge dominating set in any port-numbered \( d \)-regular graph. Apply \( A \) to \( G \); let \( D \) be the edge dominating set produced by \( A \). Consider an \( \ell \in \{ 1, 2, \ldots, d \} \). To dominate \( S(\ell) \), there must exist a node \( a_{\ell,i} \in A(\ell) \) that is incident to an edge \( e \in D \); in particular, the output \( X(a_{\ell,i}) \) is non-empty. Since \( f(a_{\ell,i}) = x_\ell \), we conclude that all nodes \( v \in V_{H(\ell)} \) in \( S(\ell) \) incident to port number \( d \) to the port 2 of another node; since \( H(\ell) \) has \( 2d - 1 \) nodes, the 2-factor contains \( 2d - 1 \) edges. This way we can find \( d \) disjoint sets of edges that are contained in \( D \), one for each \( \ell \), and each set consists of \( 2d - 1 \) edges; hence \( |D| \geq (2d - 1)d \). We conclude that the approximation ratio of \( A \) is at least
\[
\frac{|D|}{|D'|} \geq \frac{(2d - 1)d}{(k+1)d} = 4k + 1 \quad k + 1,
\]
which completes the proof of Theorem 2.

5. DISTINGUISHABLE NEIGHBOURS

This section introduces concepts and lemmas that are used in Sections 6 and 7 to facilitate algorithm design. Throughout this section, let \( G \) be a simple port-numbered graph. Then for each edge \( \{v, u\} \in E_G \) there are unique port numbers \( i \) and \( j \) such that \( p_G(v, i) = (u, j) \); we use the notation \( \ell_G(v, u) = i \) and \( \ell_G(u, v) = j \) to refer to these port numbers.

The label pair of an edge \( \{v, u\} \in E_G \) is the unordered pair \( \ell_G(v, u) = \{ \ell_G(v, u), \ell_G(u, v) \} \). The set of uniquely labelled edges of \( v \in V_G \) consists of the edges incident to \( v \) whose label pair is different from the label pair of any other edge incident to \( v \). We say that the node \( u \) is the distinguishable neighbour of \( v \in V_G \) if \( \{v, u\} \in E_G \) is the uniquely labelled edge of \( v \) that minimises the port number \( \ell_G(v, u) \).

Whenever a node has at least one uniquely labelled edge, then it also has exactly one distinguishable neighbour. For example, in the graph \( H \) of Figure 2, \( a \) is the distinguishable neighbour of \( b \), and \( d \) is the distinguishable neighbour of \( c \). However, the node \( a \) does not have any uniquely labelled edges, and hence it does not have a distinguishable neighbour, either. A key observation is that this can happen only if the node has an even degree (see Figure 8a for an example of a 3-regular graph: all nodes have distinguishable neighbours).

**Lemma 1.** Let \( v \in V_G \) be a node with an odd degree. Then the node \( v \) has a distinguishable neighbour.

**Proof.** For all \( i \) and \( j \), there are at most two edges incident to \( v \) with the label pair \( \{i, j\} \): one connected to the port \( i \) and the other connected to the port \( j \). Discard such pairs of edges with duplicate label pairs; since \( \ell_G(v) \) is odd, at least one edge with a unique label pair is retained.

Let \( M_G(i, j) \) consist of all edges \( \{v, u\} \in G \) such that \( p_G(v, i) = (u, j) \) and \( u \) is the distinguishable neighbour of \( v \); see Figure 8b for an illustration.

**Lemma 2.** For all \( i \) and \( j \), \( M_G(i, j) \) is a matching in \( G \).

**Proof.** To reach a contradiction, assume that \( \{v, t\} \) and \( \{v, u\} \) are two distinct but adjacent edges in \( M_G(i, j) \) for some \( i, j \). We must have \( \ell_G(v, t) \neq \ell_G(v, u) \) and \( \ell_G(v, t) \neq \ell_G(v, u) \); in particular, \( i \neq j \). W.l.o.g., let \( \ell_G(v, t) = i \) and \( \ell_G(v, u) = j \). From the definition of \( M_G(i, j) \), it follows that \( t \) has to be the distinguishable neighbour of \( v \), and \( v \) has to be the distinguishable neighbour of \( u \); moreover, \( \ell_G(v, t) = \ell_G(v, u) = \{i, j\} \). However, then \( \{v, t\} \) cannot be a uniquely labelled edge of \( v \), and \( t \) cannot be the distinguishable neighbour of \( v \).

Note that the sets \( M_G(i, j) \) can be constructed by a distributed algorithm in constant time. To rephrase Lemmas 1 and 2, we can construct a collection of matchings whose union covers all nodes with an odd degree. Note that the matchings \( M_G(i, j) \) are not necessarily disjoint; we may have \( i \neq j \) and \( M_G(i, j) \cap M_G(j, i) \neq \emptyset \).
6. OPTIMAL ALGORITHMS FOR REGULAR GRAPHS

Let us first present a trivial algorithm that shows that the lower bound of Theorem 1 is tight.

**Theorem 3.** There is a deterministic distributed $O(1)$-time algorithm that finds a factor $4 - 2/d$ approximation of a minimum edge dominating set in any $d$-regular port-numbered graph for any $d = 1, 2, \ldots$.

**Proof.** The algorithm outputs all edges that are connected to a port with port number 1.

Let $D$ be the output of the algorithm in a port-numbered graph $G$. First observe that $D$ is a feasible solution, as it covers all nodes and hence dominates all edges. To analyse the approximation ratio, note that the number of edges in the solution $D$ is at most $|V_G|$. Since the graph is $d$-regular, we have $d|V_G| = 2|E_G|$. Each edge in an optimal solution $D^*$ dominates at most $2d - 1$ edges, i.e., $|E_G| \leq (2d - 1)|D^*|$. Thus the approximation factor is $|D|/|D^*| \leq 4 - 2/d$. $\square$

The following result shows that the lower bound of Theorem 2 is tight as well.

**Theorem 4.** There is a deterministic distributed $O(d^2)$-time algorithm that finds a factor $4 - 6/(d + 1)$ approximation of a minimum edge dominating set in any $d$-regular port-numbered graph for any $d = 1, 2, \ldots$.

**Proof.** Let $G$ be a $d$-regular port-numbered graph. The algorithm constructs an edge dominating set $D \subseteq E_G$ in two phases, both of which can be implemented in $O(d^2)$ communication rounds. Initially, set $D \leftarrow \emptyset$.

In phase I, we consider each pair $(i, j)$ with $i, j \in \{1, 2, \ldots, d\}$ sequentially (in an arbitrary order), and for each pair $(i, j)$ we process all distinguishable edges $e \in M_G(i, j)$ in parallel: if both endpoints of $e$ are already covered by $D$, we ignore $e$, otherwise we add $e$ to $D$. See Figure 8c for an example.

In phase II, we consider again each pair $(i, j)$ sequentially, and for each pair $(i, j)$ we process all edges $e \in D \cap M_G(i, j)$ in parallel: if both endpoints of $e$ are covered by $D \setminus \{e\}$, remove $e$ from $D$. Finally, the algorithm outputs the set $D$; see Figure 8d for an example.

Recall that each set $M_G(i, j)$ is a matching; hence the decisions related to the edges in $M_G(i, j)$ are independent of each other and can be performed in parallel. Phase I constructs a spanning forest $D$: The set $D$ covers the same set of nodes as the union of the sets $M_G(i, j)$, as we ignore only redundant edges; therefore $D$ is an edge cover. Moreover, we never add edges that could close a cycle; hence the subgraph induced by $D$ is a forest.

Phase II removes some redundant edges from $D$: the property that $D$ is an edge cover is preserved throughout phase II. Moreover, phase II guarantees that there cannot be a path of length 3 in the forest $D$: if there is a path with three edges, both endpoints of the middle edge are covered by other edges. Thus $D$ is a forest of node-disjoint stars. In particular, each tree in $D$ contains at most $d$ edges, and therefore $|D| \leq d|V_G|/(d + 1)$.

It follows that the set $D$ is a feasible solution – an edge cover is an edge dominating set. Moreover,

$$|D| \leq \frac{d}{d + 1}|V_G| = \frac{2}{d + 1}|E_G| \leq \frac{4d - 2}{d + 1}|D^*|. \quad \square$$

7. OPTIMAL ALGORITHMS FOR BOUNDED-DEGREE GRAPHS

So far we have discussed distributed algorithms for regular graphs; now we turn our attention to bounded-degree graphs. Throughout this section $\Delta$ is a positive integer.
Let $A$ be a family of algorithms parametrised by $\Delta$, and let $\alpha$ be a real-valued function of $\Delta$. We say that $A$ finds an $\alpha$-approximation for edge dominating set in bounded-degree graphs if the following holds for every $\Delta$: if $G$ is a port-numbered graph such that $d_G(v) \leq \Delta$ for all $v \in V_G$, then the algorithm $A(\Delta)$ finds an $\alpha(\Delta)$-approximation of a minimum edge dominating set in $G$.

Obviously $\alpha(\Delta + 1) \geq \alpha(\Delta) \geq 1$, and $k$-regular graphs satisfy $d_G(v) \leq k$ by definition. Hence Theorem 1 has the following corollary.

**Corollary 1.** For any family of algorithms that finds an $\alpha$-approximation for edge dominating set in bounded-degree graphs, we have $\alpha(1) \geq 1$ and $\alpha(2k + 1) \geq \alpha(2k) \geq 4 - 1/k$ for all $k = 1, 2, \ldots$.

In this section we show that the corollary is tight.

**Theorem 5.** There is a family of algorithms that finds an $\alpha$-approximation for edge dominating set in bounded-degree graphs such that $\alpha(1) = 1$ and $\alpha(2k + 1) = \alpha(2k) = 4 - 1/k$ for all $k = 1, 2, \ldots$. The running time of $A(\Delta)$ is $O(\Delta^2)$.

The case $\Delta = 1$ is trivial: the optimal edge dominating set consists of all edges. In what follows, let $k \geq 1$ and $\Delta = 2k + 1$. We present the algorithm $A(\Delta)$ and show that $\alpha(\Delta) \leq 4 - 1/k$; the claim $\alpha(2k) \leq 4 - 1/k$ then follows by choosing $A(2k) = A(2k + 1)$.

### 7.1 Algorithm

In the algorithm, we will construct two node-disjoint sets of edges: a matching $M$ and a 2-matching $P$. Initially, set $M \leftarrow \emptyset$, and $P \leftarrow \emptyset$. Refer to Figure 9 for an illustration.

In phase I, we consider each edge $(i, j)$ with $i, j \in \{1, 2, \ldots, \Delta\}$ sequentially, and for each pair $(i, j)$ we process all distinguishable edges $e = \{u, v\} \in M_0(i, j)$ in parallel: if neither $u$ nor $v$ is covered by $M$, we add $e$ to $M$. This phase requires $O(\Delta^2)$ synchronous communication rounds.

In phase II, we consider each $i \in \{2, 3, \ldots, \Delta\}$ sequentially. Let $B_i$ consist of the edges $\{u, v\} \in E_G$ such that $d_G(u) < d_G(v) = i$ and neither $u$ nor $v$ is covered by $M$. The subgraph of $G$ induced by $B_i$ is bipartite; we can easily 2-colour it by assigning the black colour to each node $v$ with $d_G(v) = i$ and the white colour to each node $u$ with $d_G(u) < i$. Hence we can also find a maximal matching $M_i$ in this subgraph in $O(i)$ rounds [13]:

- Each black node sends proposals to its white neighbours, in the order of increasing port numbers, until a proposal is accepted or the list of white neighbours is exhausted.
- Each white node accepts the first proposal it gets (if any), breaking the ties with its port numbers.

Each edge with an accepted proposal is added to $M_i$. After constructing $M_i$, set $M \leftarrow M \cup M_i$, and proceed with the next value of $i$. In total, phase II requires $O(\Delta^2)$ rounds.

In phase III, we consider the subgraph $H$ induced by the edges that are not yet covered by $M$. We find a 2-matching $P$ in $H$ that dominates all edges in $H$. This is possible by using a simple $O(\Delta)$-time algorithm [21] that constructs the bipartite double cover $\mathcal{H}'$ of $H$, finds a maximal matching in the bipartite graph $\mathcal{H}'$, and maps the matching back to the original graph $H$. Informally, the algorithm proceeds as follows:

- On odd rounds, each node sends proposals to its neighbours, in the order of increasing port numbers, until a proposal is accepted or the list of neighbours is exhausted.
- On even rounds, each node receives proposals and accepts the first proposal it gets (if any), breaking the ties with its port numbers.

Each edge with an accepted proposal is added to $M_i$. After constructing $M_i$, set $M \leftarrow M \cup M_i$, and proceed with the next value of $i$. In total, phase II requires $O(\Delta^2)$ rounds.

### 7.2 Feasibility

By construction, the set $D$ dominates all edges. To see this, let $(v, u) \in E_G$ be an edge that is not dominated by $M$: hence it is part of the subgraph $H$ that we consider in Phase III. If $v$ does not send a proposal to $u$, then $v$ must have received an acceptance earlier and $v$ is covered.
by \( P \). Otherwise \( u \) receives at least one proposal and hence becomes covered by \( P \).

### 7.3 Properties

Let us then proceed to analyse the approximation factor. In the analysis, we need the following properties of \( M \) and \( P \).

(a) The sets \( M \) and \( P \) are node-disjoint, \( M \) is a matching, and \( P \) is a 2-matching in \( G \).

(b) If \( v \in V_G \) has an odd degree, then \( v \) is covered by \( M \), or there is a neighbour \( u \) of \( v \) that is covered by \( M \).

(c) If \( \{u, v\} \in P \) then \( d_G(v) = d_G(u) \).

The algorithm clearly preserves property (a). Property (b) follows from phase I. To verify property (c), observe that if \( d_G(v) \neq d_G(u) \), we would have covered \( v \) or \( u \) in phase II.

### 7.4 Definitions

For a set \( X \subseteq E_G \) of edges, we say that a node \( v \in V_G \) is an \( X \)-node if it is covered by \( X \). Hence each node is an \( M \)-node, a \( P \)-node, or neither.

By property (b) above, we can construct a set \( C \subseteq E_G \) of edges such that (i) each edge \( e \in C \) joins a \( P \)-node and an \( M \)-node, and (ii) each \( P \)-node with an odd degree is incident to exactly one edge in \( C \). Note that \( M \), \( P \), and \( C \) are disjoint subsets of \( E_G \). Define \( F = E_G \setminus (M \cup P \cup C) \).

Now let \( D^* \) be an arbitrary maximal matching in \( G \); in particular, \( D^* \) can be a minimum maximal matching and hence a minimum edge dominating set for \( G \) (recall Section 1.1). We proceed to show that \( |D| \) is not too large in comparison with \( |D^*| \).

Each node covered by \( D^* \) is called an internal node, and all other nodes are called external nodes, or there is one internal node \( u \) with \( \{u, v\} \in P \), and finally \( d_G(v) - 3 \) other nodes \( x_1, x_2, \ldots, x_3 \) \( \in F \). By property (c), we have \( d_G(v) = d_G(s) = d_G(t) \). Hence the weights of the edges are \( d_G(v) = (4 - d_G(v)) + 2 - d_G(v) = 2 - d_G(v) \).

For a set \( F \subseteq E_G \), we use this observation in a double-counting argument.

### 7.7 Double Counting

Let \( v \) be an external \( P \)-node. There are at most 2 edges in \( P \) that are incident to \( v \), and hence at least 2 \( d_G(v) - 2 \) edges in \( F \cup C \) that are incident to \( v \). Hence the total weight is \( w(v) \geq 0 \). In particular, the total weight of all edges in the graph is non-negative.

In the following, we consider an internal node \( v \), and derive an upper bound on \( w(v) \) as a function of \( c(v) \).

If \( c(v) = 2 \), then \( v \) has to be incident to two edges in \( D \), and these edges have to join \( v \) and an external node. Hence the neighbours of \( v \) can be classified as follows: there are two external nodes \( s \) and \( t \), with \( \{s, t\} \in P \), such that \( v \in N_G(v) \). There is one internal node \( u \) with \( \{v, u\} \in P \), and \( v \in N_G(v) \).

By property (c), we have \( d_G(v) = d_G(s) = d_G(t) \). Hence the weights of the edges are \( w(v, s) = w(v, t) = 2 - d_G(v) \).

Let \( w(v, u) = 0 \), and \( w(v, x) \in [0, 2] \) for each \( i \), depending on whether \( x_i \) happens to be an external \( P \)-node. It follows that the total weight of incident edges is

\[
|D(v)| \leq (2 - d_G(v)) + (|D(v)| - 3)|2| = 2.
\]

If \( c(v) = 3/2 \), then \( v \) has to have two neighbours, an external \( P \)-node \( s \) and an internal \( P \)-node \( t \) such that \( \{s, t\} \in P \) and \( \{v, t\} \in P \). Again, \( d_G(v) = d_G(s) \), and hence we have assigned the weight \( w(v, s) = 2 - d_G(v) \).

There are two sub-cases. First, if \( d_G(v) = \Delta \), then \( v \) is also adjacent to an \( M \)-node \( u \) such that \( \{v, u\} \in C \); by the choice of \( w \), we have \( w(v, u) = 0 \). In addition to \( s \), \( t \), and \( u \), there are at least \( \Delta - 3 \) other nodes \( x_1, x_2, \ldots, x_3 \) adjacent to \( v \); we have \( \{v, x_i\} \in F \) and hence \( w(v, x_i) \leq 2 \). It follows that the total weight of incident edges is

\[
|D(v)| \leq 2 \Delta + (2 + (\Delta - 3)|2| = 2 \Delta - 3.
\]

Otherwise \( d_G(v) \leq \Delta - 1 \), and \( v \) is adjacent to \( d_G(v) - 2 \) other nodes \( x_1, x_2, \ldots, x_3 \) in addition to \( v \); we have \( \{v, x_i\} \in F \) and hence \( w(v, x_i) \leq 2 \). It follows that the total weight is

\[
|D(v)| \leq 2 - d_G(v) + (d_G(v) - 2)|2| = \Delta - 3.
\]

If \( c(v) = 1 \), we always have at least two edges incident to \( v \) with a non-positive weight: if \( v \) is incident to two edges in \( D \), then we have two internal \( P \)-nodes \( s \) and \( t \) with \( \{v, s\} \in P \) and \( \{v, t\} \in P \); since each of \( s \), \( t \), and \( u \) is internal, these edges have zero weight. Otherwise \( v \) is incident to only one edge in \( D \), let it be \( \{v, s\} \in D \). In that case \( s \) has to be an external node; hence there has to be another internal node \( t \) with \( \{v, t\} \in D^* \). The weight of \( \{v, t\} \) is zero, as it joins a pair of internal nodes, and the weight of \( \{v, s\} \) is \( 2 - d_G(v) \leq 0 \) if \( \{v, s\} \in P \) and zero if \( \{v, s\} \in M \). We conclude that \( v \) is incident to at most \( d_G(v) - 2 \) edges with a positive weight; since the weight of any edge is at most 2, we have the upper bound

\[
|D(v)| \leq (d_G(v) - 2)|2| = 2 \Delta - 4.
\]
Finally, if $c(v) \leq 1/2$, it is sufficient to note that $v$ is adjacent to another internal node $u$ with $\{v, u\} \in D^*$, and $w(\{v, u\}) = 0$. Hence there are at most $d_G(v) - 1$ edges incident to $v$ with a positive weight; we have the upper bound
\[ w(v) \leq (d_G(v) - 1)2 \leq 2\Delta - 2. \]

Summing over all internal nodes $i \in I$, we can find the following upper bound on the total weight of all edges:
\[ W = -2I_4 + (\Delta - 3)I_3 + (2\Delta - 4)I_2 + (2\Delta - 2)I_1 + (2\Delta - 2)I_0. \]
Since $0 \leq w(E_G) \leq W$, we have $W \geq 0$ and
\[ 2I_4 \leq (\Delta - 3)I_3 + (2\Delta - 4)I_2 + (2\Delta - 2)I_1 + (2\Delta - 2)I_0. \]

### 7.8 Approximation Ratio

Now we are ready to derive an upper bound on the approximation ratio of the algorithm $A(\Delta)$:
\[
\frac{|D|}{|D^*|} = \frac{4I_4 + 3I_3 + 2I_2 + I_1}{I_4 + I_3 + I_2 + I_1 + I_0} \\
= 4 - \frac{2I_4 + 4I_3 + 6I_2 + 8I_1 + 8I_0}{2I_4 + 2I_3 + 2I_2 + I_1 + 2I_0} \\
\leq 4 - \frac{2I_4 + 4I_3 + 6I_2 + 8I_1 + 8I_0}{(\Delta - 1)I_4 + (2\Delta - 2)I_2 + 2\Delta I_1 + 2\Delta I_0} \\
\leq 4 - \frac{2}{\Delta - 1} = 4 - \frac{1}{k},
\]
Here we have used the assumption that $\Delta \geq 3$ and hence $3/\Delta \geq 2/(\Delta - 1)$. We conclude that $\alpha(\Delta) \leq 4 - 1/k$, and Theorem 5 follows.

### 8. ACKNOWLEDGEMENTS

Thanks to Matti Åstrand, Patrik Florén, Petteri Kaski, Topi Musto, Valentin Polishchuk, Joel Rybicki, and Jara Uitto for discussions, and to anonymous reviewers for their comments. This work was supported in part by the Academy of Finland, Grant 132380.

### 9. REFERENCES


