Stable Approximation Algorithms for Dominating Set and Independent Set

Mark de Berg
Department of Mathematics and Computer Science, TU Eindhoven, The Netherlands

Arpan Sadhukhan
Department of Mathematics and Computer Science, TU Eindhoven, The Netherlands

Frits Spieksma
Department of Mathematics and Computer Science, TU Eindhoven, The Netherlands

Abstract

We study Dominating Set and Independent Set for dynamic graphs in the vertex-arrival model. We say that a dynamic algorithm for one of these problems is *k-stable* when it makes at most k changes to its output independent set or dominating set upon the arrival of each vertex. We study trade-offs between the stability parameter k of the algorithm and the approximation ratio it achieves. We obtain the following results.

- We show that there is a constant $\varepsilon^* > 0$ such that any dynamic $(1+\varepsilon^*)$-approximation algorithm for Dominating Set has stability parameter $\Omega(n)$, even for bipartite graphs of maximum degree 4.
- We present algorithms with very small stability parameters for Dominating Set in the setting where the arrival degree of each vertex is upper bounded by d. In particular, we give a 1-stable $(d+1)^2$-approximation, and a 3-stable $(9d/2)$-approximation algorithm.
- We show that there is a constant $\varepsilon^* > 0$ such that any dynamic $(1+\varepsilon^*)$-approximation algorithm for Independent Set has stability parameter $\Omega(n)$, even for bipartite graphs of maximum degree 3.
- Finally, we present a 2-stable $O(d)$-approximation algorithm for Independent Set, in the setting where the average degree of the graph is upper bounded by some constant d at all times.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Mathematics of computing → Graph algorithms

Keywords and phrases Dynamic algorithms, approximation algorithms, stability, dominating set, independent set

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2023.27

Category APPROX

Funding MdB, AS, and FS are supported by the Dutch Research Council (NWO) through Gravitation-grant NETWORKS-024.002.003.

1 Introduction

Given a simple, undirected graph $G = (V, E)$, a *dominating set* is a subset $D \subset V$ such that each vertex in V is either a neighbor of a vertex in D, or it is in D itself. An *independent set* is a set of vertices $I \subset V$ such that no two vertices in I are neighbors. Dominating Set (the problem of finding a minimum-size dominating set) and Independent Set (the problem of finding a maximum-size independent set) are fundamental problems in algorithmic graph theory. They have numerous applications and served as prototypical problems for many algorithmic paradigms.

We are interested in Dominating Set and Independent Set in a dynamic setting, where the graph G changes over time. In particular, we consider the well-known *vertex-arrival model.* Here one starts with an empty graph $G(0)$, and new vertices arrive one by one,

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2023).
Editors: Nicole Megow and Adam D. Smith; Article No. 27; pp. 27:1–27:19
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

along with their edges to previously arrived vertices. In this way, we obtain a sequence of graphs $G(t)$, for $t = 0, 1, 2, \ldots$. Our algorithm is then required to maintain a valid solution – a dominating set, or an independent set – at all times. In the setting we have in mind, computing a new solution is not the bottleneck, but each change to the solution (adding or deleting a vertex from the solution) is expensive. Of course we also want that the maintained solution has a good approximation ratio. To formalize this, and following De Berg et al. [12], we say that a dynamic algorithm is a *k-stable ρ-approximation algorithm* if, upon the arrival upon each vertex, the number of changes (vertex additions or removals) to the solution is at most k and the solution is a ρ-approximation at all times. In this framework, we study trade-offs between the stability parameter k and the approximation ratio that can be achieved. Ideally, we would like to have a so-called *stable approximation scheme (SAS)*: an algorithm that, for any given yet fixed parameter $\varepsilon > 0$ is k_ε-stable and gives a $(1+\varepsilon)$ approximation algorithm, where k_ε only depends on ε and not on the size of the current instance. (There is an intimate relation between local-search PTASs and SASs; we come back to this issue in Section 2.)

The vertex-arrival model is a standard model for online graph algorithms, and our stability framework is closely related to online algorithms with bounded recourse. However, there are two important differences. First, computation time is free in our framework – for instance, the algorithm may decide to compute an optimal solution in exponential time upon each insertion – while in online algorithms with bounded recourse the update time is typically taken into account. Thus we can fully focus on the the trade-off between stability and approximation ratio. Secondly, we consider the approximation ratio of the solution, while in online algorithm one typically considers the competitive ratio. Thus we compare the quality of our solution at time t to the *static optimum*, which is simply the optimum for the graph $G(t)$. A competitive analysis, one the other hand, compares the quality of the solution at time t to the *offline optimum*: the best solution for $G(t)$ that can be computed by a dynamic algorithm that knows the sequence $G(0), \ldots, G(t)$ in advance but must still process the sequence with bounded recourse. See also the discussion in the paper by Boyar et al. [6]. Thus approximation ratio is a much stronger notion that competitive ratio. As a case in point, consider Dominating Set, and suppose that n singleton vertices arrive, followed by a single vertex with edges to all previous ones. Then the static optimum for the final graph is 1, while the offline optimum with bounded recourse is $\Omega(n)$.

Related work. We now review some of the most relevant existing literature on the online version of our problems. (Borodin and El-Yaniv [5] give a general introduction to online computation.) The classical online model, where a vertex that has been added to the dominating set or to the independent set, can never be removed from it – that is, the no-recourse setting – is e.g. considered by King and Tzeng [18] and Lipton and Tomkins [20]. They show that already for the special case of interval graphs no online algorithm has constant competitive ratio; see also De et al. [9], who study these two problems for geometric intersection graphs. For Dominating Set in the vertex-arrival model, Boyar et al. [6] give online algorithms with bounded competitive ratio for trees, bipartite graphs, bounded degree graphs, and planar graphs. They actually analyze both the competitive ratio and the approximation ratio (which, as discussed earlier, can be quite different). A crucial difference between the work of Boyar et al. [6] and ours is that they do not allow recourse: in their setting, once a vertex is added to the dominating set, it cannot be removed.

To better understand online algorithmic behavior, various ways to relax classical online models have been studied. In particular, for Independent Set, among others, Halldórsson et al. [16] consider the option for the online algorithm to maintain multiple solutions. Göbel et al. [13] analyze a stochastic setting where, among other variants, a randomly generated graph is presented in adversarial order (the prophet inequality model), and one where an adversarial graph is presented in random order. In these settings they find constant competitive algorithms for Independent Set on interval graphs, and more generally for graphs with bounded so-called inductive independence number.

One other key relaxation of the online model is to allow recourse. Having recourse can be seen as relaxing the irrevocability assumption in classical online problems, and allows to assess the impact of this assumption on the competitive ratio, see Boyar et al. [7]. The notion of (bounded) recourse has been investigated for a large set of problems. Without aiming for completeness, we mention Angelopoulos et al. [1] and Gupta et al. [15] who deal with online matching and matching assignments, Gupta et al. [14] who investigate the set cover problem, and Berndt et al. [4] who deal with online bin covering with bounded migration, and Berndt et al. [3] who propose a general framework for dynamic packing problems. For these problems, it is shown what competitive or approximation ratios can be achieved when allowing a certain amount of recourse (or migration). Notice that, in many cases, an amortized interpretation of recourse is used; then the average number of changes to a solution is bounded (instead of the maximum, as for k-stable algorithms defined above). For instance, Lsiu and Charington-Toole [21] show that, for independent set, there is an interesting trade-off between the competitive ratio and the amortized recourse cost: for any $t > 1$, they provide a t-competitive algorithm for independent set using $t - 1$ recourse cost. Their results however do not apply to our notion of stability.

Our contribution. We obtain the following results.

- In Section 2, we show that the existence of a local-search PTAS for the static version of a graph problem, implies, under certain conditions, the existence of a SAS for the problem in the vertex-arrival model (whereas the converse need not be true). This implies that for graphs with strongly sublinear separators, a SAS exists for Independent Set and for Dominating Set when the arrival degrees – that is, the degrees of the vertices upon arrival – are bounded by a constant.
- In Section 3, we consider Dominating Set in the vertex-arrival model. Let d denote the maximum arrival degree. We show (i) there does not exist a SAS even for bipartite graphs of maximum degree 4, (ii) there is a 1-stable $(d + 1)^2$-approximation algorithm, and (iii) there is a 3-stable $\frac{9d}{2}$-approximation algorithm.
- In Section 4, we consider Independent Set in the vertex-arrival model. We show that there does not exist a SAS even for bipartite graphs of maximum degree 3. Further, we give a 2-stable $O(d)$-approximation algorithm for the case where the average degree of $G(t)$ is bounded by d at all times.

2 Stable Approximation Schemes versus PTAS by local search

In this section we discuss the relation between Stable Approximation Schemes (SASs) and Polynomial-Time Approximation Schemes (PTASs). Using known results on local-search PTASs, we then obtain SASs for Independent Set for Dominating Set on certain graph classes. While the results in this section are simple, they set the stage for our main results in the next sections.

The goals of a SAS and a PTAS are the same: both aim to achieve an approximation ratio $(1 + \varepsilon)$, for any given $\varepsilon > 0$. A SAS, however, works in a dynamic setting with the requirement that k_ε, the number of changes per update, is a constant for fixed ε, while a

PTAS works in a static setting with the condition that the running time is polynomial for fixed ε. Hence, there are problems that admit a PTAS (or are even polynomial-time solvable) but no SAS [12].

One may think that the converse should be true: if a dynamic problem admits a SAS, then its static version admits a PTAS. Indeed, we can insert the input elements one by one and let the SAS maintain a $(1+\varepsilon)$-approximation, by performing at most k_ε changes per update. For a SAS, there is no restriction on the time needed to update the solution, but it seems we can simply try all possible combinations of at most k_ε changes in, say, $n^{O(k_\varepsilon)}$ time. This need not work, however, since there could be many ways to update the solution using at most k_ε changes. Even though we can find all possible combinations in polynomial time, we may not be able to decide which combination is the right one: the update giving the best solution at this moment may get us stuck in the long run. The SAS can avoid this by spending exponential time to decide what the right update is. Thus the fact that a problem admits a SAS does not imply that it admits a PTAS.

Notwithstanding the above, there is a close connection between SASs and PTASs and, in particular between SASs and so-called local-search PTASs: under certain conditions, the existence of a local-search PTAS implies the existence of a SAS. For simplicity we will describe this for graph problems, but the technique may be applied to other problems.

Let $G = (V, E)$ be a graph and suppose we wish to select a minimum-size (or maximum-size) subset $S \subset V$ satisfying a certain property. Problems of this type include Dominating Set, Independent Set, Vertex Cover, Feedback Vertex Set, and more. A local-search PTAS for such a graph minimization problem starts with an arbitrary feasible solution S – the whole vertex set V, say – and then it tries to repeatedly decrease the size of S by replacing a subset $S_{\text{old}} \subset S$ by a subset $S_{\text{new}} \subset V \setminus S$ such that $|S_{\text{new}}| = |S_{\text{old}}| - 1$ and $(S \setminus S_{\text{old}}) \cup S_{\text{new}}$ is still feasible. (For a maximization problem we require $|S_{\text{new}}| = |S_{\text{old}}| + 1$.) This continues until no such replacement can be found.[1] A key step in the analysis of a local-search PTAS is to show the following, where n is the number of vertices.

- **Local-Search Property.** If S is a feasible solution that is not a $(1+\varepsilon)$-approximation then there are subsets $S_{\text{old}}, S_{\text{new}}$ as above with $|S_{\text{old}}| \leqslant f_\varepsilon$, for some f_ε depending only on ε and not on n.

This condition indeed gives a PTAS, because we can simply try all possible pairs $S_{\text{old}}, S_{\text{new}}$, of which there are $O(n^{2f_\varepsilon})$.

Now consider a problem that has the Local-Search Property in the vertex-arrival model, possibly with some extra constraint (for example, on the arrival degrees of the vertices). Let $G(t)$ denote the graph after the arrival of the t-th vertex, and let $\text{OPT}(t)$ denote the size of an optimal solution for for $G(t)$. We can obtain a SAS if the problem under consideration has the following properties.

- **Continuity Property.** We say that the dynamic problem (in the vertex-arrival model, possibly with extra constraints) is *d-continuous* if $|\text{OPT}(t+1) - \text{OPT}(t)| \leqslant d$. In other words, the size of an optimal solution should not change by more than d when a new vertex arrives. Note that the solution itself may change completely; we only require its size not to change by more than d.
- **Feasibility Property.** For maximization problems we require that any feasible solution for $G(t-1)$ is also a feasible solution for $G(t)$, and for minimization problems we require that any feasible solution for $G(t-1)$ can be turned into a feasible solution for $G(t)$ by

[1] See the paper by Antunes etal [2] for a nice exposition on problems solved using local-search PTAS.

adding the arriving vertex to the solution. (This condition can be relaxed to saying that we can repair feasibility by $O(1)$ modifications to the current solution, but for concreteness we stick to the simpler formulation above.)

Note that Independent Set, Vertex Cover, and Feedback Vertex Set are 1-continuous, and that Dominating Set is $(d-1)$-continuous when the arrival degree of the vertices is bounded by $d \geqslant 2$. Moreover, these problems all have the Feasibility Property.

For problems that have the Local-Search Property as well as the Continuity and Feasibility Properties, it is easy to give a SAS. Hence, non-existence of SAS for a problem with Continuity and Feasibility directly implies non-existence of local-search PTAS.[2] We give the pseudocode for minimization problems, but for maximization problems a similar approach works.

Algorithm 1 SAS-for-Continuous-Problems(v).

1: ▷ v is the vertex arriving at time t
2: $S_{\text{alg}} \leftarrow S_{\text{alg}}(t-1) \cup \{v\}$ ▷ S_{alg} is feasible for $G(t)$ by the feasibility condition
3: **while** S_{alg} is not a $(1+\varepsilon)$-approximation **do**
4: Find sets $S_{\text{old}} \subset S_{\text{alg}}$ and $S_{\text{new}} \subset V(t) \setminus S_{\text{alg}}$ with $|S_{\text{old}}| \leqslant f_\varepsilon$ and $|S_{\text{new}}| = |S_{\text{old}}| - 1$ such that $(S_{\text{alg}} \setminus S_{\text{old}}) \cup S_{\text{new}}$ is a valid solution, and set $S_{\text{alg}} \leftarrow (S_{\text{alg}} \setminus S_{\text{old}}) \cup S_{\text{new}}$.
5: $S_{\text{alg}}(t) \leftarrow S_{\text{alg}}$

▶ **Theorem 1.** *Any graph problem that has the Continuity Property, the Feasibility Property and the Local-Search Property admits a SAS in the vertex arrival model, with stability parameter $(d+1)\cdot(2f_\varepsilon - 1)+1$ for minimization problems and $d \cdot (2f_\varepsilon + 1)$ for maximization problems.*

Proof. First consider a minimization problem. By the Feasibility Property and the working of the algorithm, the solution that SAS-for-Continuous-Problems computes upon the arrival of a new vertex is feasible. Moreover, it must end with a $(1+\varepsilon)$-approximation because of the Local-Search Property. Before the while-loop we add v to S_{alg}, and in each iteration of the while-loop, at most f_ε vertices are deleted from S_{alg} and at most $f_\varepsilon - 1$ vertices are added. We claim that the number of iterations is at most $\lceil (1+\varepsilon)d \rceil + 1$. Indeed, before the arrival of v we have $|S_{\text{alg}}(t-1)| \leqslant (1+\varepsilon)\cdot|\textsc{opt}(t-1)|$, and we have $|\textsc{opt}(t-1) - \textsc{opt}(t)| \leqslant d$ by the Local-Search Property. Hence, after $\lceil (1+\varepsilon)d \rceil + 1$ iterations we have

$$\begin{aligned}
|S_{\text{alg}}| &= |S_{\text{alg}}(t-1)| + 1 - (\lceil (1+\varepsilon)d \rceil + 1) \\
&\leqslant (1+\varepsilon)\cdot|\textsc{opt}(t-1)| - (1+\varepsilon)d \\
&\leqslant (1+\varepsilon)\cdot(\textsc{opt}(t) + d) - (1+\varepsilon)d \\
&\leqslant (1+\varepsilon)\cdot\textsc{opt}(t).
\end{aligned}$$

For a maximization problem the proof is similar. The differences are that we do not add an extra vertex to S_{alg} in step 2, that the number of changes per iteration is at most $2f_\varepsilon + 1$, and that the number of iterations is at most d. (We do not get the factor $(1+\varepsilon)$ because if opt increases, then the error that we can make, which is $\varepsilon \cdot \textsc{opt}$, also increases.) ◀

This general result allows us to obtain a SAS for a variety of problems, for graph classes for which a local-search PTAS is known.

[2] Dominating Set and Independent Set, even with maximum degree bounded by 3, do not admit a PTAS assuming P $\neq$ NP [8, 22]. In sections 3.1 and 4.1, by proving the non-existence of a SAS for Dominating Set with maximum degree bounded by 4 and Independent Set with maximum degree bounded by 3, we thus show non-existence of local-search PTAS, independent of the assumption of P $\neq$ NP.

Recall that a balanced *separator* of a graph $G = (V, E)$ with n vertices is a subset $S \subset V$ such that $V \setminus S$ can be partitioned into subsets A and B with $|A| \leqslant 2n/3$ and $|B| \leqslant 2n/3$ and no edges between A and B. We say that a graph class[3] $\mathcal{G}$ has *strongly sublinear separators*, if any graph $G \in \mathcal{G}$ has a balanced separator of size $O(n^\delta)$, for some fixed constant $\delta < 1$. Planar graphs, for instance, have separators of size $O(\sqrt{n})$ [19]. A recent generalization of separators are so-called *clique-based separators*, which are separators that consist of cliques and whose size is measured in terms of the number of cliques [10]. (Actually, the cost of a separator S that is the union of cliques $C_1, \ldots, C_k$ is defined as $\sum_{i=1}^{k} \log(|C_i| + 1)$, but this refined measure is not needed here.) Disk graphs (which do not have normal separators of sublinear size) have a clique-based separator of size $O(\sqrt{n})$, for instance, and pseudo-disk graphs have a clique-based separator of size $O(n^{2/3})$ [11]. For graph classes with strongly sublinear separators there are local-search PTASs for several problems. Combining that with the technique above gives the following result.

▶ **Corollary 2.** *The following problems admit a SAS in the vertex-arrival model.*

(i) Independent Set *on graph classes with sublinear clique-based separators.*

(ii) Dominating Set *on graph class with sublinear separators, when the arrival degree of each vertex bounded by some fixed constant d.*

Proof. As noted earlier, Independent Set is 1-continuous and Dominating Set is $(d-1)$-continuous. Moreover, these problems have the Feasibility Property. It remains to check the Local-Search Property.

(i) Any graph class with a separator of size $O(n^\delta)$ has the Local-Search Property for Independent Set; see the paper by Her-Peled and Quanrud [17]. (In that paper they show the Local-Search Property for graphs of polynomial expansion – see Corollary 26 and Theorem 3.4 – and graphs of polynomial expansion have sublinear separators.) Theorem 1 thus implies the result for such graph classes. To extend this to clique-based separators, we note that (for Independent Set) we only need the Local-Search Property for graphs that are the union of two independent sets, namely the independent set S and an optimal independent set S_{opt}. Such graphs are bipartite, so the largest clique has size two. Hence, the existence of a clique-based separator of size $O(n^\delta)$ immediately implies the existence of a normal separator of size $O(n^\delta)$.

(ii) For Dominating Set on graphs with polynomial expansion (hence, on graphs with sublinear separators) the Local-Search Property holds [17, Theorem 3.15]. Theorem 1 thus implies an $O(d \cdot f_\varepsilon)$-stable $(1 + \varepsilon)$-approximation algorithm, for some constant f_ε depending only on ε. Hence, if d is a fixed constant , we obtain a SAS. ◀

3 Dominating Set

In this section we study stable approximation algorithms for Dominating Set in the vertex arrival model. We first show that the problem does not admit a SAS, even when the maximum degree of the graph is bounded by 4. After that we will describe two algorithms that achieve constant approximation ratio with constant stability, in the setting where each vertex arrives with constant degree.

Let $G = (V, E)$ be a graph. For a subset $S \subset V$, we denote the open neighborhood of a subset $W \subset V$ in G by $N_G(W)$, so $N_G(W) := \{v \in V \setminus W : \text{there is a } w \in W \text{ with } (v, w) \in E\}$. The closed neighborhood $N_G[W]$ is defined as $N_G(W) \cup W$. When the graph G is clear from the context, we may omit the subscript G and simply write $N(W)$ and $N[W]$.

[3] We only consider hereditary graph classes, that is, graph classes $\mathcal{G}$ such that any induced subgraph of a graph in $\mathcal{G}$ is also in $\mathcal{G}$.

3.1 No SAS for graphs of maximum degree 4

Our lower-bound construction showing that Dominating Set does not admit a SAS – in fact, our construction will show a much stronger result, namely that there is a fixed constant $\varepsilon^* > 0$ such that any stable $(1+\varepsilon^*)$-approximation algorithm must have stability $\Omega(n)$ – is based on a certain type of expander graphs, as given by the following proposition. Note that L (the left part of the bipartition of the vertex set) is larger by a constant fraction than R (the right part of the bipartition), while the expansion property goes from L to R. The proof of the following proposition was communicated to us by Noga Alon.

▶ **Proposition 3.** *For any $\mu > 0$ and any n that is sufficiently large, there are constants $0 < \varepsilon, \delta < 1$ such that there is a bipartite graph $G_{\text{exp}}(L \cup R, E)$ with the following properties:*
- *$|L| = (1+\varepsilon)n$ and $|R| = n$.*
- *The degree of every vertex in G is at most 3.*
- *For any $S \subset L$ with $|S| \leqslant \delta n$ we have $|N(S)| \geqslant (2-2\mu)|S|$*

Proof. Let $\mu > 0$ and let t be an integer so that $t > 1/\mu$. Let $\varepsilon \leqslant 1/3^{2t+1}$ be a fixed positive number. Let $H = (A \cup B, E_H)$ be a 3-regular bipartite graph with vertex classes A and B, each of size $(1+\varepsilon)n$, in which for every subset $S \subset B$ of size at most δn we have $|N(S)| \geqslant (2-\mu)|S|$, for some fixed real number $\delta = \delta(\mu) > 0$. (It is known that random cubic bipartite graphs have this property with high probability.) Now pick a set $T \subset A$ of εn vertices so that the distance between any pair of them is larger that $2t$. Such a set exists, as one can choose its members one by one, making sure to avoid the balls of radius $2t$ around the already chosen vertices. This gives a set T of the desired size since $\varepsilon 3^{2t+1} < 1$. We define G_{exp} to be the induced subgraph of H on the classes of vertices $R = A \setminus T$ and $L = B$. It remains to show that G_{exp} has the desired properties.

We have $|L| = (1+\varepsilon)n$ and $|R| = n$ by construction, and the maximum degree of G_{exp} is clearly at most 3. Now let S be a set of at most δn vertices in L. We have to show that $N_{G_{\text{exp}}}(S)$, its neighbor set in G_{exp}, has size at least $(2-2\mu)|S|$. We can assume without loss of generality that $S \cup N_{G_{\text{exp}}}(S)$ is connected in G_{exp}, since we can apply the bound to each connected component separately and just add the inequalities. Note that this assumption implies that $S \cup N_H(S)$ is also connected in H. Observe that the neighborhood $N_H(S)$ of S in the original graph H, which is contained in B, is of size at least $(2-\mu)|S|$, by the property of H. If the set T of deleted vertices has at most $\mu|S|$ members in $N_H(S)$ then the desired inequality holds and we are done. Otherwise T has more than $\mu|S|$ vertices that belong to $N_H(S)$. But the distance between any two such vertices in H is larger than $2t$ (and so the balls of radius t around them are disjoint), and since $S \cup N_H(S)$ is connected this would imply $|S| > t|T \cap N_H(S)| > t\mu|S| > |S|$, which is a contradiction. ◀

Now consider Dominating Set for a dynamic graph in the vertex-arrival model. Let $G(t)$ denote the graph at time t, that is, after the first t insertions. Let $\varepsilon^* > 0$ be such that $\varepsilon^* < \min\left(\frac{\varepsilon}{2+\varepsilon}, \frac{0.49\delta}{2(1+\varepsilon)}\right)$, where ε and δ are the constants in the expander construction of Proposition 3. Consider a dynamic algorithm ALG for Dominating Set such that $|D_{\text{alg}}(t)| \leqslant (1+\varepsilon^*) \cdot \text{OPT}(t)$ at any time t, where $D_{\text{alg}}(t)$ is the output dominating set of ALG at time t and $\text{OPT}(t)$ is the minimum size of a dominating set for $G(t)$. Let $f_{\varepsilon^*}(n)$ denote the stability of ALG, that is, the maximum number of changes it performs on D_{alg} when a new vertex arrives, where n is the number of vertices before the arrival.

We now give a construction showing that, for arbitrarily large n, there is a sequence of n arrivals that requires $f_{\varepsilon^*}(n) \geqslant \frac{1}{6(7+6\varepsilon)}\lfloor \delta n \rfloor$. To this end, choose N large enough such that the bipartite expander graph $G_{\text{exp}} = (L \cup R, E)$ from Proposition 3 exists for $\mu = 0.005$ and

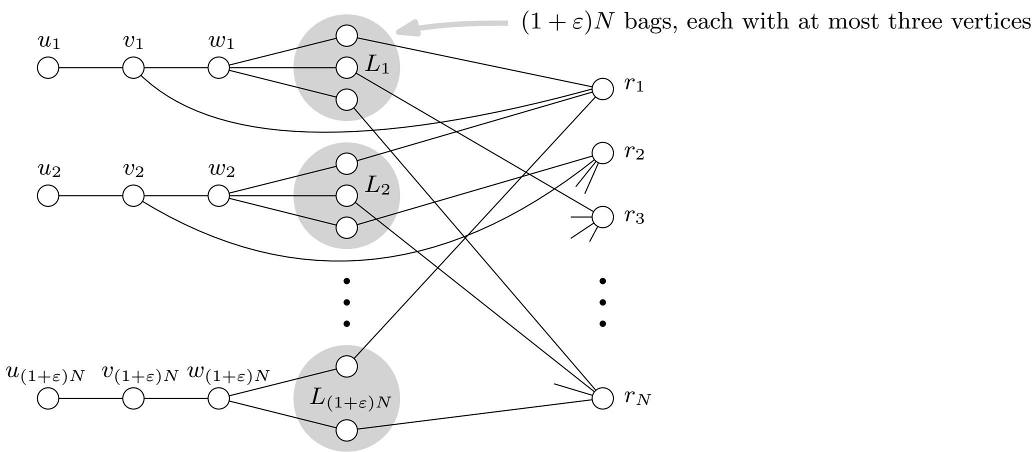

Figure 1 The lower-bound construction for DOMINATING SET.

$|R| = N$. Label the vertices in L as $\ell_1, \ldots, \ell_{|L|}$ and the vertices in R as $r_1, \ldots, r_{|R|}$. Our construction uses five layers of vertices, arriving one by one, as described next and illustrated in Fig. 1.

- *Layer 1:* The first layer consists of vertices $u_1, \ldots, u_{(1+\varepsilon)N}$, each arriving as a singleton.
- *Layer 2:* The second layer consists of vertices $v_1, \ldots, v_{(1+\varepsilon)N}$, where each v_i has an edge to vertex u_i from the first layer.
- *Layer 2:* The third layer consists of vertices $w_1, \ldots, w_{(1+\varepsilon)N}$, where each w_i has an edge to v_i from the second layer.
- *Layer 4:* Let $G_{\text{exp}} = (L \cup R, E)$ be the expander from Proposition 3. The fourth layer consists of $|L| = (1+\varepsilon)N$ bags, each with at most three vertices. More precisely, if $\deg(\ell_i)$ is the degree of vertex $\ell_i \in L$ in the expander G_{exp}, then bag L_i has $\deg(\ell_i)$ vertices. Each vertex in L_i has an edge to vertex w_i from the third layer.
- *Layer 5:* Finally, the fifth layer arrives. Each vertex in this layer corresponds to a vertex r_i from the bipartite expander G_{exp} and, with a slight abuse of notation, we will also denote it by r_i. If, in $G\,\text{exp}$, r_i has an edge to some vertex ℓ_j in G_{exp}, then the corresponding vertex r_i in our construction will have an edge to some vertex in the bag L_j. Clearly, we can do this in such a way that each vertex in any of the bags L_i has an edge to exactly one vertex r_i. In addition to the edges to (vertices in) the bags, each vertex r_i also has an edge to the vertex v_i from the second layer.

Let t_1 be the time at which the last vertex of $L_{(1+\varepsilon)N}$ was inserted, and let t_2 be the time at which r_N was inserted. Let $G(t)$ denote the graph induced by all vertices inserted up to time t. Thus $G(t_1)$ consists of the layers 1–4, and $G(t_2)$ consists of layers 1–5.

▶ **Observation 4.** *For any t with $t_1 \leqslant t \leqslant t_2$ we have $\text{OPT}(t) \leqslant (2+2\varepsilon)N$. Moreover, $\text{OPT}(t_2) \leqslant (2+\varepsilon)N$.*

Proof. For any $t_1 \leqslant t \leqslant t_2$, the set $D_1 := \{v_1, \ldots, v_{(1+\varepsilon)N}\} \cup \{w_1, \ldots, w_{(1+\varepsilon)N}\}$ forms a dominating set for $G(t)$. Moreover, $D_1 := \{v_1, \ldots, v_{(1+\varepsilon)N}\} \cup \{r_1, \ldots, r_N\}$ forms a dominating set for $G(t_2)$. ◀

We call a bag L_i *fully dominated* by a set D of vertices if each vertex in L_i is dominated by some vertex in D. Observation 4 states that $\text{OPT}(t_2)$ is significantly smaller than $\text{OPT}(t_1)$, which is because the vertices in R can fully dominate all bags. This means that $D_{\text{alg}}(t_2)$ must contain most vertices of R, in order to achieve the desired approximation. Adding only a few vertices from R will be too expensive, however, since fully dominating a small number of bags will be expensive, because of the expander property of G_{exp}. Hence, if the stability parameter $f_{\varepsilon^*}(n)$ is small, then ALG cannot maintain the desired approximation ratio. Next we make this proof idea precise.

▶ **Lemma 5.** *Let $D_{\mathrm{alg}}(t_2)$ denote the output dominating set for $G(t_2)$. Suppose $D_{\mathrm{alg}}(t_2) \cap R$ fully dominates at most δN bags L_i. Then $|D_{\mathrm{alg}}(t_2)| > (1+\varepsilon^*) \cdot \mathrm{OPT}(t_2)$.*

Proof. Let $m < \delta N$ denote the number of bags fully dominated by $D_{\mathrm{alg}}(t_2) \cap R$. Consider a bag L_i that is not fully dominated by $D_{\mathrm{alg}}(t_2) \cap R$. Then $D_{\mathrm{alg}}(t_2)$ must contain the vertex w_i or at least one vertex from the bag L_i. Hence, the number of vertices in $D_{\mathrm{alg}}(t_2)$ from the third and fourth layer is at least $(1+\varepsilon)N - m$. Moreover, $D_{\mathrm{alg}}(t_2)$ must have at least $(1+\varepsilon)N$ vertices from the first and second layer, to dominate all vertices from the first layer. Observe that in order to fully dominate a bag L_i by vertices in R, we need all vertices in R with an edge to some vertex of L_i. So if $D_{\mathrm{alg}}(t_2) \cap R$ fully dominates $m \leqslant \delta N$ bags, then $|D_{\mathrm{alg}}(t_2) \cap R| \geqslant 1.99m$ by the properties of the expander graph G_{exp} in Proposition 3. Hence,

$$|D_{\mathrm{alg}}(t_2)| > (1+\varepsilon)N - m + (1+\varepsilon)N + 1.99m \geqslant (2+2\varepsilon)N.$$

Observation 4 thus implies that $\frac{|D_{\mathrm{alg}}(t_2)|}{\mathrm{OPT}(t_2)} \geqslant \frac{2+2\varepsilon}{2+\varepsilon} > 1+\varepsilon^*$. ◀

Lemma 5 means that, in order to achieve approximation ratio $1+\varepsilon^*$, the set $D_{\mathrm{alg}}(t_2) \cap R$ must fully dominate more than δN bags. Next we show that this cannot be done when the stability parameter $f_{\varepsilon^*}(n)$ is $o(n)$.

▶ **Lemma 6.** *Let t^* be the first time when $D_{\mathrm{alg}}(t^*) \cap R$ fully dominates at least δN bags. If $f_{\varepsilon^*}(n) < \frac{1}{6(7+6\varepsilon)} \lfloor \delta n \rfloor$ then $|D_{\mathrm{alg}}(t^*)| > (1+\varepsilon^*) \cdot \mathrm{OPT}(t^*)$.*

Proof. Let n_t denote the number of vertices of the graph $G(t)$, and observe that $n_{t^*} \leqslant (7+6\varepsilon)N$. Hence, $f_{\varepsilon^*}(n_{t^*}) \leqslant \frac{1}{6}\delta N$. By definition of t^*, we know that just before time t^* the set $D_{\mathrm{alg}} \cap R$ fully dominates less than δN bags. Because ALG is $f_{\varepsilon^*}(n)$-stable, the number of vertices from R added to D_{alg} at time t^* is at most $f_{\varepsilon^*}(n_{t^*})$. Since these new vertices have degree at most three, they can complete the full domination of at most $3f_{\varepsilon^*}(n_{t^*})$ bags. Thus,

$$\big(\text{ number of bags fully dominated by } D_{\mathrm{alg}}(t^*) \cap R\,\big) \quad < \quad \delta N + 3f_{\varepsilon^*}(n_{t^*}) \quad \leqslant \quad \frac{3}{2}\delta N.$$

Let L_i be a bag that is not fully dominated by $D_{\mathrm{alg}}(t^*) \cap R$. Since $D_{\mathrm{alg}}(t^*)$ is a dominating set, it must then contain the vertex w_i or at least one vertex from L_i. Hence, the number of vertices in $D_{\mathrm{alg}}(t^*)$ from layers 3 and 4 is more than $(1+\varepsilon)N - \frac{3}{2}\delta N$. In addition, $D_{\mathrm{alg}}(t^*)$ must have at least $(1+\varepsilon)N$ vertices from layers 1 and 2. Finally, in order to fully dominate a bag L_i by vertices in R, we need all the vertices in R that have an edge to some vertex of L_i. In other words, $D_{\mathrm{alg}}(t^*) \cap R$ must contain all neighbors of the fully dominated bags. Since $D_{\mathrm{alg}}(t^*) \cap R$ dominates at least δN bags, we know that $|D_{\mathrm{alg}}(t^*) \cap R| \geqslant 1.99 \cdot \delta N$, by the properties of the expander graph in Proposition 3. Hence,

$$\begin{aligned}|D_{\mathrm{alg}}(t^*)| &> (1+\varepsilon)N - \frac{3}{2}\delta N + (1+\varepsilon)N + 1.99 \cdot \delta N \\ &\geqslant \left(1 + \frac{0.49\delta}{2+2\varepsilon}\right)(2+2\varepsilon)N \\ &\geqslant (1+\varepsilon^*) \cdot \mathrm{OPT}(t^*).\end{aligned}$$ ◀

By Lemmas 5 and 6 we obtain the following result.

▶ **Theorem 7.** *There is a constant $\varepsilon^* > 0$ such that any dynamic $(1+\varepsilon^*)$-approximation algorithm for* DOMINATING SET *in the vertex arrival model, must have stability parameter $\Omega(n)$, even when the maximum degree of any of the graphs $G(t)$ is bounded by 4.*

3.2 Constant-stability algorithms when the arrival degrees are bounded

In the previous section we saw that there is no SAS for Dominating Set, even when the maximum degree is bounded by 4. In this section we present stable algorithms whose approximation ratio depends on the arrival degree of the vertices. More precisely, we give a simple 1-stable algorithm with approximation ratio $(d+1)^2$ and a more complicated 3-stable algorithm with approximation ratio $9d/2$, where d is the maximum degree of any vertex upon arrival. Note we only restrict the degree upon arrival: the degree of a vertex may further increase due to the arrival of new vertices. This implies that deletions cannot be handled by a stable algorithm with bounded approximation ratio: even with arrival degree 1 we may create a star graph of arbitrarily large size, and deleting the center of the star cannot be handled in a stable manner without compromising the approximation ratio.

A 1-stable $(d+1)^2$-approximation algorithm. Recall that $G(t)$ denotes the graph after the arrival of the t-th vertex. We can turn $G(t)$ into a directed graph $\vec{G}(t)$, by directing each edge towards the older of its two incident vertices. In other words, when a new vertex arrives then its incident edges are directed away from it.

Let $N^+[v] := \{v\} \cup \{\text{out-neighbors of } v \text{ in } \vec{G}(t)\}$, where t is such that v is inserted at time t. In other words, $N^+[v]$ contains v itself plus the neighbors of v immediately after its arrival. Let $\text{OPT}_{\text{out}}(t)$ denote the minimum size of a dominating set in $\vec{G}(t)$ under the condition that every vertex v should be dominated by a vertex in $N^+[v]$. We call such a dominating set a *directed dominating set*. Note that a directed dominating set for $\vec{G}(t)$ is a dominating set in $G(t)$ as well. The following lemma states that $\text{OPT}_{\text{out}}(t)$ is not much larger than $\text{OPT}(t)$.

▶ **Lemma 8.** *At any time t we have $\text{OPT}_{\text{out}}(t) \leqslant (d+1) \cdot \text{OPT}(t)$.*

Proof. Let $D_{\text{opt}}(t)$ be a minimum dominating set for $G(t)$. Let $D := \bigcup_{v \in D_{\text{opt}}(t)} N^+[v]$. Observe that D is a directed dominating set for $\vec{G}(t)$. Since every vertex arrives with degree at most d, we have $|N^+[v]| = d+1$. The result follows. ◀

We call two vertices u, v *unrelated* if $N^+[u] \cap N^+[v] = \emptyset$, otherwise u, v are *related*. The following lemma follows immediately from the definition of $\text{OPT}_{\text{out}}(t)$.

▶ **Lemma 9.** *Let $U(t)$ be a set of pairwise unrelated vertices in $\vec{G}(t)$. Then $\text{OPT}_{\text{out}}(t) \geqslant |U(t)|$.*

Our algorithm will maintain a directed dominating set $D_{\text{alg}}(t)$ for $\vec{G}(t)$ and a set $U(t)$ of pairwise unrelated vert. Since the initial graph is empty, we initialize $D_{\text{alg}}(0) := \emptyset$ and $U(0) := \emptyset$. When a new vertex v arrives at time t, we proceed as follows.

Algorithm 2 Directed-DomSet(v).

```
▷ v is the vertex arriving at time t
if N⁺[v] ∩ D_alg(t−1) ≠ ∅ then                     ▷ v is already dominated
    Set D_alg(t) ← D_alg(t−1) and U(t) ← U(t−1)
else
    if v is unrelated to all vertices u ∈ U(t−1) then
        Set U(t) ← U(t−1) ∪ {v} and D_alg(t) ← D_alg(t−1) ∪ {v}
    else
        Let u be a vertex related to v, that is, with N⁺[u] ∩ N⁺[v] ≠ ∅.
        Pick an arbitrary vertex w ∈ N⁺[u] ∩ N⁺[v].
        Set D_alg(t) ← D_alg(t−1) ∪ {w} and U(t) ← U(t−1).
```

This leads to the following theorem.

▶ **Theorem 10.** *There is a* 1*-stable* $(d+1)^2$*-approximation algorithm for* Dominating Set *in the vertex-arrival model, where* d *is the maximum arrival degree of any vertex.*

Proof. Clearly the set D_{alg} maintained by Directed-DomSet is a directed dominating set. Moreover the algorithm is 1-stable as after each arrival, it either adds a single vertex to D_{alg} or does nothing. It is easily checked that the set U maintained by the algorithm is always a set of pairwise unrelated vertices, and that all vertices in D_{alg} are an out-neighbor of a vertex in U or are in U themselves. Hence, by Lemmas 8 and 9, at any time t we have

$$|D_{\text{alg}}(t)| \leqslant (d+1) \cdot |U(t)| \leqslant (d+1) \cdot \text{OPT}_{\text{out}}(t) \leqslant (d+1)^2 \cdot \text{OPT}(t),$$

which finishes the proof. ◀

In Appendix A we show that the approximation ratio $\Theta(d^2)$ is tight for this algorithm.

A 3-stable $(9d/2)$-approximation algorithm. The algorithm presented above has optimal stability, but its approximation ratio is $\Omega(d^2)$. We now present an algorithm whose stability is still very small, namely 3, but whose approximation ratio is only $O(d)$. This is asymptotically optimal, since, as is easy to see, any algorithm with constant stability must have approximation ratio $\Omega(d)$. Our approach is somewhat similar to that of Liu and Toole-Charignon [21], but a key difference is that we obtain a worst-case bound on the stability and they obtain an amortized bound. Our algorithm works in *phases*, as explained next. Suppose we start a new phase at time t and let $D_{\text{alg}}(t-1)$ be the output dominating set at time $t-1$. The algorithm then computes a minimum dominating set $D_{\text{opt}}(t)$ for the graph $G(t)$, which we call the *target dominating set*. The algorithm will then slowly migrate from $D_{\text{alg}}(t-1)$ to $D_{\text{opt}}(t)$, by first adding the vertices in $D^+ := D_{\text{opt}}(t) \setminus D_{\text{alg}}(t-1)$ and then removing the vertices in $D^- := D_{\text{alg}}(t-1) \setminus D_{\text{opt}}(t)$. This is done in $\lceil |D^+ \cup D^-|/2 \rceil$ steps. Vertices that arrive in the meantime are also added to the dominating set, to ensure that the output remains a dominating set at all time. After all vertices in D^+ and D^- have been added and deleted, respectively, the next phase starts. Next we describe and analyze the algorithm in detail.

At the start of the whole algorithm, at time $t = 0$, we initialize $D_{\text{alg}}(0) := \emptyset$, and $D(0)^+ = \emptyset$ and $D(0)^- = \emptyset$.

Algorithm 3 Set-and-Achieve-Target(v).

▷ v is the vertex arriving at time t and $G(t)$ is the graph after arrival of v
$D_{\text{alg}} \leftarrow D_{\text{alg}}(t-1) \cup \{v\}$
if $D^+(t-1) = \emptyset$ and $D^-(t-1) = \emptyset$ **then** ▷ start a new phase
  Let $D_{\text{opt}}(t)$ be a minimum dominating set for $G(t)$.
  Set $D^+(t) \leftarrow D_{\text{opt}}(t) \setminus D_{\text{alg}}(t-1)$ and $D^-(t) \leftarrow D_{\text{alg}}(t-1) \setminus D_{\text{opt}}(t)$.
else
  Set $D^+(t) \leftarrow D^+(t-1)$ and $D^-(t) \leftarrow D^-(t-1)$
Set $m^+ \leftarrow \min(2, |D^+(t)|)$. Delete m^+ vertices from $D^+(t)$ and add them to D_{alg}.
Set $m^- \leftarrow \min(2-m^+, |D^-(t)|)$. Delete m^- vertices from $D^-(t)$ and delete the same vertices from D_{alg}.
$D_{\text{alg}}(t) \leftarrow D_{\text{alg}}$

The algorithm defined above is 3-stable, as it adds one vertex to D_{alg} in step 2 and then makes two more changes to D_{alg} in steps 8 and 9. Next we prove that its approximation ratio is bounded by $9d/2$. Note that the size of a minimum dominating set can reduce over time, due to the arrival of new vertices. The next lemma shows that this reduction is bounded. Let $\text{MAX-OPT}(t) := \max(\text{OPT}(1), \text{OPT}(2), \ldots, \text{OPT}(t))$ denote the maximum size of any of the optimal solutions until (and including) time t.

▶ **Lemma 11.** *For any time t we have $\text{MAX-OPT}(t) \leqslant d \cdot \text{OPT}(t)$, where d is the maximum arrival degree of any vertex.*

Proof. Let $t^* \leqslant t$ be such that $\text{MAX-OPT}(t) = \text{OPT}(t^*)$. Let $D_{\text{opt}}(t)$ be an optimal dominating set for $G(t)$ and define $V(t^*)$ to be the set of vertices of $G(t^*)$. Let D be the set of vertices in $D_{\text{opt}}(t)$ that were not yet present at time t^*, and define $D^* := (D_{\text{opt}}(t) \setminus D) \cup N_{t^*}(D)$, where $N_{t^*}(D)$ contains the neighbors of D in $V(t^*)$. Then D^* is a dominating set for $G(t^*)$ since any vertex in $V(t^*)$ that is not dominated by a vertex in $D_{\text{opt}}(t^*) \setminus D$ is in D^* itself. Moreover, $|D^*| \leqslant d \cdot \text{OPT}(t)$, since each vertex in D has at most d neighbors in $V(t^*)$. ◀

We first bound the size of D_{alg} at the start of each phase. Note that in the proofs below, $D^+(t)$ and $D^-(t)$ refer to the situation before the execution of line 8 and 9 in the algorithm SET-AND-ACHIEVE-TARGET.

▶ **Lemma 12.** *If a new phase starts at time t, then $D_{\text{alg}}(t-1) \leqslant 3 \cdot \text{MAX-OPT}(t-1)$.*

Proof. We proceed by induction on t. The lemma trivially holds at the start of the first phase, when $t = 1$. Now consider the start of some later phase, at time t, and let t_{prev} be the previous time at which a new phase started. Recall that $D_{\text{alg}}(t) = D_{\text{opt}}(t_{\text{prev}}) \cup \{\text{vertices arriving at times } t_{\text{prev}}, t_{\text{prev}}+1, \ldots, t-1\}$.

Moreover,

$$|D^+(t_{\text{prev}}) \cup D^-(t_{\text{prev}})| \leqslant |D_{\text{alg}}(t_{\text{prev}}-1)| + |D_{\text{opt}}(t_{\text{prev}})| \leqslant 3 \cdot \text{MAX-OPT}(t_{\text{prev}}-1) + \text{OPT}(t_{\text{prev}}),$$

where the last inequality uses the induction hypothesis. From time t_{prev} up to time $t-1$, the vertices from $D^+(t_{\text{prev}}) \cup D^-(t_{\text{prev}})$ are added/deleted in pairs, so

$$t - t_{\text{prev}} = \left\lceil \frac{3 \cdot \text{MAX-OPT}(t_{\text{prev}}-1) + \text{OPT}(t_{\text{prev}})}{2} \right\rceil \leqslant \left\lceil \frac{4 \cdot \text{MAX-OPT}(t_{\text{prev}})}{2} \right\rceil = 2 \cdot \text{MAX-OPT}(t_{\text{prev}}).$$

Hence,

$$\begin{aligned} D_{\text{alg}}(t-1) &\leqslant \text{OPT}(t_{\text{prev}}) + (t - t_{\text{prev}}) \\ &\leqslant \text{MAX-OPT}(t_{\text{prev}}) + 2 \cdot \text{MAX-OPT}(t_{\text{prev}}) \\ &\leqslant 3 \cdot \text{MAX-OPT}(t-1) \end{aligned}$$ ◀

The previous lemma bounds $|D_{\text{alg}}|$ just before the start of each phase. Next we use this to bound $|D_{\text{alg}}|$ during each phase.

▶ **Lemma 13.** *For any time t we have $|D_{\text{alg}}(t)| \leqslant (9/2) \cdot \text{MAX-OPT}(t)$.*

Proof. Consider a time t. If a new phase starts at time $t+1$ then the lemma follows directly from Lemma 12. Otherwise, let $t_{\text{prev}} \leqslant t$ be the last time at which a new phase started, and let t_{next} be the next time at which a new phase starts. Furthermore, let $t^* := \max\{t' : t_{\text{prev}} \leqslant t' < t_{\text{next}} \text{ and } D^+(t') \neq \emptyset\}$. In other words, t^* is the last time step in the interval $[t_{\text{prev}}, t_{\text{next}})$ at which we still add vertices from D^+ to D_{alg}. If $D^+(t_{\text{prev}})$ is empty then let $t^* = t_{\text{prev}}$. It is easy to see from the algorithm that $|D_{\text{alg}}(t)| \leqslant |D_{\text{alg}}(t^*)|$. Note that $D_{\text{alg}}(t^*)$ contains the vertices from $D_{\text{alg}}(t_{\text{prev}} - 1)$, plus the vertices from $D_{\text{opt}}(t_{\text{prev}})$, plus the vertices that arrived from time t_{prev} to time t^*. Hence,

$$\begin{aligned}
|D_{\text{alg}}(t^*)| &\leqslant |D_{\text{alg}}(t_{\text{prev}} - 1)| + \text{OPT}(t_{\text{prev}}) + (t^* - t_{\text{prev}} + 1) \\
&\leqslant |D_{\text{alg}}(t_{\text{prev}} - 1)| + \text{OPT}(t_{\text{prev}}) + \frac{\text{OPT}(t_{\text{prev}})}{2} \\
&\leqslant 3 \cdot \text{MAX-OPT}(t_{\text{prev}} - 1) + \text{MAX-OPT}(t_{\text{prev}}) + \frac{\text{OPT}(t_{\text{prev}})}{2} \\
&\leqslant (9/2) \cdot \text{MAX-OPT}(t).
\end{aligned}$$

Note that from the first to the second line we replaced $(t^* - t_{\text{prev}} + 1)$ by $\text{OPT}(t_{\text{prev}})/2$, which we can do because we add vertices from $D_{\text{opt}}(t_{\text{prev}})$ in pairs. It may seem that we should actually write $\left\lceil \frac{\text{OPT}(t_{\text{prev}})}{2} \right\rceil$ here. When $\text{OPT}(t_{\text{prev}})$ is odd, however, then the algorithm can already remove a vertex in D^- from D_{alg} when the last vertex from D^+ is added to D_{alg}. (This is not true in the special case when $D^-(t_{\text{prev}}) = \emptyset$, but in that case the second term is an over-estimation. Indeed, when $D^-(t_{\text{prev}}) = \emptyset$ then $D_{\text{alg}}(t_{\text{prev}} - 1) \subseteq D_{\text{opt}}(t_{\text{prev}})$ and so $|D^+(t_{\text{prev}})| = |D_{\text{opt}}(t_{\text{prev}}) \setminus D_{\text{alg}}(t_{\text{prev}} - 1)| < |\text{OPT}(t_{\text{prev}} - 1)|$.) This finishes the proof. ◀

Putting Lemmas 11 and 13 together, we obtain the following theorem.

▶ **Theorem 14.** *There is a 3-stable $(9d/2)$-approximation algorithm for* Dominating Set *in the vertex-arrival model, where d is the maximum arrival degree of any vertex.*

4 Stable Approximation Algorithms for Independent Set

In this section we first show that Independent Set does not admit a SAS, even when restricted to graphs of maximum degree 3. Then we give an 2-stable $O(d)$-approximation algorithm for graphs whose average degree is bounded by d.

4.1 No SAS for graphs of maximum degree 3

We prove our no-SAS result for Independent Set in a similar (but simpler) way as for Dominating Set. Thus we actually prove the stronger result that there is a constant $\varepsilon^* > 0$ such that any dynamic $(1+\varepsilon^*)$-approximation algorithm for Independent Set in the vertex arrival model, must have stability parameter $\Omega(n)$, in this case even when the maximum degree of any of the graphs $G(t)$ is bounded by 3.

Let $\varepsilon^* > 0$ be a real number less than $\min\left(\frac{0.82\delta}{1-0.82\delta}, \varepsilon\right)$. Let ALG be an algorithm that maintains an independent set I_{alg} such that $\text{OPT}(t) \leqslant (1+\varepsilon^*) \cdot |I_{\text{alg}}(t)|$ at all times. Let $f_{\varepsilon^*}(n)$ denote the stability of ALG, that is, the maximum number of changes it performs on I_{alg} when a new vertex arrives, where n is the number of vertices before the arrival. We will show that, for arbitrarily large n, there is a sequence of n arrivals that requires $f_{\varepsilon^*}(n) \geqslant \frac{1}{6(2+\varepsilon)}\lfloor \delta n \rfloor$. As before, choose N large enough such that the bipartite expander graph $G_{\text{exp}} = (L \cup R, E)$ from Proposition 3 exists for $\mu = 0.005$ and $|R| = N$. In our no-SAS construction for Independent Set, we only need G_{exp}, not additional layers are needed. Thus the construction is simply as follows.

- First the vertices $r_1, \ldots, r_N$ from the set R arrive one by one, as singletons.
- Next the vertices $\ell_1, \ldots, \ell_{(1+\varepsilon)N}$ from L arrive one by one (in any order), along with their incident edges in G_{exp}.

▶ **Lemma 15.** *Let t^* be the first time when $|I_{\text{alg}}(t) \cap L| \geqslant \delta N$. If $f_{\varepsilon^*}(n) < \frac{1}{6(2+\varepsilon)}\lfloor \delta n \rfloor$ then $\text{OPT}(t^*) > (1+\varepsilon^*) \cdot |I_{\text{alg}}(t^*)|$.*

Proof. Let n_t denote the number of vertices of the graph $G(t)$, and observe that $n_{t^*} \leqslant (2+\varepsilon)N$. Hence, $f_{\varepsilon^*}(n_{t^*}) \leqslant \frac{1}{6}\delta N$. By definition of t^*, we know that just before time t^* we have $|I_{\text{alg}} \cap L| < \delta N$. Because ALG is $f_{\varepsilon^*}(n)$-stable, we have

$$|I_{\text{alg}}(t^*) \cap L| \leqslant \delta N + f(n_{t_f}) \leqslant \frac{7}{6}\delta N.$$

Since $\mu = 0.005$, from Proposition 3 we have $|N(I_{\text{alg}}(t^*) \cap L| \geqslant 1.99 \cdot \delta N$. Hence $|I_{\text{alg}}(t^*) \cap R| \leqslant N - 1.99 \cdot \delta N$, and so

$$|I_{\text{alg}}(t^*)| = |I_{\text{alg}}(t^*) \cap R| + |I_{\text{alg}}(t^*) \cap L| \leqslant \frac{7}{6}\delta N + N - 1.99 \cdot \delta N < N - 0.82\,\delta N$$

. We also have $|\text{OPT}(t^*)| \geqslant N$. Hence, $\text{OPT}(t^*) > \frac{N}{N-0.82\delta N}|I_{\text{alg}}(t^*)| > (1+\varepsilon^*) \cdot |I_{\text{alg}}(t^*)|$. ◀

▶ **Theorem 16.** *There is a constant $\varepsilon^* > 0$ such that any dynamic $(1+\varepsilon^*)$-approximation algorithm for* INDEPENDENT SET *in the vertex arrival model, must have stability parameter $\Omega(n)$, even when the maximum degree of any of the graphs $G(t)$ is bounded by 3.*

Proof. By Proposition 3 the maximum degree of the graph is always bounded by 3. Let $t = 2N + \varepsilon N$ and let $|I_{\text{alg}}(t) \cap L| = M$. We know by Lemma 15 that if ALG is a $(1+\varepsilon^*)$-approximation and if $f_{\varepsilon^*}(n) < \frac{1}{6(2+\varepsilon)}\lfloor \delta n \rfloor$ then $M \leqslant \delta n$. Hence $|I_{\text{alg}}(t) \cap R| \leqslant N - 1.99M$. So we have

$$|I_{\text{alg}}(t)| = |I_{\text{alg}}(t) \cap R| + |I_{\text{alg}}(t) \cap L| \leqslant N - 1.99M + M \leqslant N.$$

But we have $\text{OPT}(t) = (1+\varepsilon)N$. Hence the approximation ratio at time $t = 2N + \varepsilon N$ is greater than or equal to $\frac{(1+\varepsilon)N}{N} = 1 + \varepsilon > 1 + \varepsilon^*$ which is a contradiction. This finishes the proof. ◀

4.2 Constant-stability algorithms when the average degree is bounded

A 2-stable $O(d)$-approximation algorithm. In this section we consider the setting where the average degree of $G(t)$ is upper bounded by some constant d at all times. It is easy to observe that in this setting, if we allow just one change after each vertex arrival, then it's not possible to get a bounded approximation ratio. However, we are able to get a bounded approximation ratio with only two changes per arrival.

It is not hard to see that if the maximum degree is bounded by some constant d^*, then a simple greedy 1-stable algorithm maintains a $O(d^*)$ approximation. Our idea is to maintain an induced subgraph with a number of vertices that is linear in the number of vertices of $G(t)$, and whose maximum degree (rather than average degree) is bounded. We then use the induced subgraph to generate an independent set. Below we make the idea precise.

First we define a (trivial) subroutine algorithm below which takes in an independent set I^* and a subset W^* of vertices as an input, and tries to add a vertex v from $W^* \setminus I^*$ to I^* such that $I^* \cup \{v\}$ is still an independent set.

Algorithm 4 GREEDY-ADDITION(I^*, W^*).

```
1: if there exist a vertex v ∈ W* \ I* such that I* ∪ {v} is an independent set then
2:     Set I* = I* ∪ {v}
```

Next we move on to describe our main algorithm, which uses GREEDY-ADDITION as a subroutine. Let $\Delta(G)$ denote the maximum degree of a graph $G = (V, E)$. For a subset $W \subset V$, define $G[W]$ to be the subgraph of G induced by W. Let $V(t)$ denote the set of vertices of $G(t)$.

Observe that by ordering the vertices of $G(t)$ in increasing order of their degree and by taking the first $\lceil \frac{99}{100}|V(t)| \rceil$ vertices, we can construct a set $V^*(t) \subseteq V$ such that $|V^*(t)| \geqslant \frac{99}{100}|V(t)|$ and $\Delta(G[V^*(t)]) \leqslant 100d$. (The number 100 has no special significance – it can be chosen much smaller – but we use it for convenience.) The idea of our algorithm is to maintain a vertex set $W(t) \subseteq V(t)$ such that $\Delta(G[W(t)]) \leqslant 100d$ and the size of $W(t)$ is linear in $|V(t)|$. In order to maintain such a subset, we work in phase, as before: at the start of each phase, the algorithm sets itself a target vertex set $V^*(t)$ of large size and with $\Delta(G[V^*(t)]) \leqslant 100d$. At time t, $W^+(t)$ and $W^-(t)$ denote the vertices that need to be added and removed respectively from $W(t)$ in order to achieve the target. This is done by the algorithm presented next, where we initialize $W^+(t), W^-(t), W(t) = \emptyset$.

Algorithm 5 SET-ACHIEVE-AND-USE-TARGET(v).

```
▷ v is the vertex arriving at time t and G(t) is the graph after arrival of v
if W⁺(t − 1) = ∅ and W⁻(t − 1) = ∅ then                ▷ start a new phase
    Choose V*(t) ⊆ V such that |V*(t)| ⩾ (99/100)|V(t)| and Δ(G[V*(t)]) ⩽ 100d.
    Set W⁺(t) ← V*(t) \ W(t − 1) and W⁻(t) ← W(t − 1) \ V*(t).
else
    Set W⁺(t) ← W⁺(t − 1) and W⁻(t) ← W⁻(t − 1)
Set m⁻ ← min(1, |W⁻(t)|). Delete m⁻ vertices from W⁻(t) and delete the same vertices
    from W(t). Call the vertex deleted(if any) as v*.
Set I_alg(t) ← I_alg(t − 1) \ {v*}
Set m⁺ ← min(1 − m⁻, |W⁺(t)|). Delete m⁺ vertices from W⁺(t) and add them to
    W(t).
GREEDY-ADDITION(I_alg(t), W(t))
```

▶ **Lemma 17.** *At the start of each phase – that is, at a time t such that $W^+(t-1) = \emptyset$ and $W^-(t-1) = \emptyset$ – we have $|W(t-1)| \geqslant \frac{495}{1000} \cdot |V(t)|$.*

Proof. We proceed by induction on t. The lemma trivially holds at the start of the first phase, when $t = 1$. Now consider the start of some later phase, at time t, and let t_{prev} be the previous time at which a new phase started. Since the start of a phase is just after the end of the previous phase we have $|W(t-1)| \geqslant \frac{99}{100}|V(t_{\text{prev}})|$.

Observe that $W^+(t_{\text{prev}})$ and $W^-(t_{\text{prev}})$ are disjoint subsets of the vertices of $V(t_{\text{prev}})$. Hence $|W^+(t_{\text{prev}}) \cup W^-(t_{\text{prev}})| \leqslant |V(t_{\text{prev}})|$. So $(t - t_{\text{prev}}) \leqslant |V(t_{\text{prev}})|$, since we make one change at a time. So $|V(t)| \leqslant 2|V(t_{\text{prev}})|$ and we have

$$|W(t-1)| \geqslant \frac{99}{100} \cdot |V(t_{\text{prev}})| \geqslant \frac{495}{1000} \cdot 2 \cdot |V(t_{\text{prev}})| \geqslant \frac{495}{1000} \cdot |V(t)|$$

This finishes the proof of the lemma. ◀

The previous lemma gives lower bound of $|W(t)|$ at the start of each phase. Next we use this to give a lower bound of $|W(t)|$ at any time point t.

▶ **Lemma 18.** *For any time t, we have $|W(t)| \geqslant \frac{485}{1010} \cdot |V(t)|$.*

Proof. Consider a time t. If a new phase starts at time t then the lemma follows directly from Lemma 17. Otherwise, let t_{prev} be the previous time at which a new phase started, and let t_{next} be the next time at which a new phase starts.

Let $t^* := \max\{t' : t_{\text{prev}} \leqslant t' < t_{\text{next}}$ and $W^-(t') \neq \emptyset\}$. In other words, t^* is the last time step in the interval $[t_{\text{prev}}, t_{\text{next}})$ at which we still delete vertices from $W(t)$. It is easy to see from the algorithm that in the interval $[t_{\text{prev}}, t_{\text{next}})$, the value $\frac{|W(t)|}{|V(t)|}$ is minimum at $t = t^*$. Observe that $|W^-(t_{\text{prev}})| \leqslant \frac{1}{100} \cdot |V(t_{\text{prev}})|$, which implies $(t^* - t_{\text{prev}}) \leqslant \frac{1}{100} \cdot |V(t_{\text{prev}})|$. Hence, $|V(t^*)| \leqslant \frac{101}{100}|V(t_{\text{prev}})|$ and so by Lemma 17 we have

$$|W(t^*)| \geqslant \frac{495}{1000} \cdot |V(t_{\text{prev}})| - \frac{1}{100} \cdot |V(t_{\text{prev}})| = \frac{485}{1010} \cdot \frac{101}{100} \cdot |V(t_{\text{prev}})| \geqslant \frac{485}{1010} \cdot |V(t^*)|$$

This finishes the proof of the lemma. ◀

▶ **Lemma 19.** *For any time t we have $\frac{|W(t)|}{|I_{\text{alg}}(t)|} \leqslant c \cdot d$ for some constant c.*

Proof. Observe that at any time point t the maximum degree of $G[W(t)]$ is bounded by $100d$. Choose a constant c such that $c \cdot d > 100d + 1$. Hence if $I \subseteq W(t)$ is an independent set with $|I| < \frac{|W(t)|}{c \cdot d}$, then there always exist a vertex $v \in W(t) \setminus I$ such that $\{v\} \cup I$ is a independent set. If not then all vertices in $W(t) \setminus I$ have an edge to I. Now $|W(t) \setminus I| \geqslant \frac{(c \cdot d - 1)}{c \cdot d}|W(t)|$, so the average degree of I in $G[W(t)]$ is greater than $c \cdot d - 1 > 100d$, which is a contradiction.

Also observe that $I_{\text{alg}}(t) \subset W(t)$. Now we proceed by induction on t. The lemma holds trivially for $t = 0$. Now suppose the lemma holds for $t = k$. There are two cases.

Case 1: At t=k+1, a vertex v^* is deleted from $W(t)$

In this case initially $I_{\text{alg}}(t) = I_{\text{alg}}(t-1) \setminus \{v^*\}$. Then the subroutine GREEDY-ADDITION tries to add a new vertex to $I_{\text{alg}}(t)$. If $\frac{|W(t)|}{|I_{\text{alg}}(t)|} \leqslant c \cdot d$ before GREEDY-ADDITION is initiated then we are done. Else by the arguments above, we know that if $|I_{\text{alg}}| < \frac{|W(t)|}{c \cdot d}$ then the subroutine GREEDY-ADDITION can always add a vertex. In that case $|I_{\text{alg}}(t)| \geqslant |I_{\text{alg}}(t-1)|$ and $|W(t)| = |W(t-1)| - 1$, so $\frac{|W(t)|}{|I_{\text{alg}}(t)|} \leqslant \frac{|W(t-1)|}{|I_{\text{alg}}(t-1)|} \leqslant c \cdot d$ by induction.

Case 2: At t=k+1, a vertex v^* is added to $W(t)$

If $\frac{|W(t)|}{|I_{\text{alg}}(t)|} \leqslant c \cdot d$ before GREEDY-ADDITION is initiated then we are done. Else we know that if $|I_{\text{alg}}| < \frac{|W(t)|}{c \cdot d}$ then the subroutine GREEDY-ADDITION can always add a vertex. In that case $|I_{\text{alg}}(t)| = |I_{\text{alg}}(t-1)| + 1$ and $|W(t)| = |W(t-1)| + 1$, so $\frac{|W(t)|}{|I_{\text{alg}}(t)|} \leqslant \frac{|W(t-1)|}{|I_{\text{alg}}(t-1)|} \leqslant c \cdot d$ by induction. ◀

▶ **Theorem 20.** *There is a 2-stable $O(d)$-approximation algorithm for* INDEPENDENT SET *in the vertex-arrival model where the average degree of $G(t)$ is bounded by d at all times.*

Proof. We know that the size of the maximum independent set at time t is trivially bounded by $|V(t)|$. Now by Lemmas 18 and 19, we have,

$$\frac{|V(t)|}{|I_{\text{alg}}(t)|} \leqslant \frac{1010}{485} \frac{|W(t)|}{|I_{\text{alg}}(t)|} \leqslant \frac{1010}{485} \cdot c \cdot d$$

Hence the algorithm SET-ACHIEVE-AND-USE-TARGET is an $O(d)$ approximation. Also observe that the maximum number of changes to $I_{\text{alg}}(t)$ occurs when there is a induced deletion of a single vertex due to the deletion of a vertex from $W(t)$ and then adding a vertex during the execution of the subroutine algorithm GREEDY-ADDITION. So clearly the stabilty of the algorithm SET-ACHIEVE-AND-USE-TARGET is bounded by 2. ◀

5 Concluding remarks

We studied the stability of dynamic algorithms for Dominating Set and Independent Set in the vertex-arrival model. For both problems we showed that a SAS does not exist. For Independent Set this even holds when the degrees of all vertices are bounded by 3 at all times. This is clearly tight, since a SAS is easily obtained on graphs of maximum degree 2. For Dominating Set the no-SAS result holds for degree-4 graphs. A challenging open problem is whether a SAS exists for Dominating Set for degree-3 graphs. We also gave algorithms whose approximation ratio and/or stability depends on the (arrival or average) degree. An interesting open problem here is: Is there a 1-stable $O(d)$-approximation algorithm for Dominating Set, when the arrival degree is at most d? Finally, we believe the concept of stability for dynamic algorithms, which purely focuses on the change in the solution (rather than computation time) is interesting to explore for other problems as well.

References

1. Spyros Angelopoulos, Christoph Dürr, and Shendan Jin. Online maximum matching with recourse. In *Proc. 43rd International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 117 of *LIPIcs*, pages 8:1–8:15, 2018. `doi:10.4230/LIPIcs.MFCS.2018.8`.
2. Daniel Antunes, Claire Mathieu, and Nabil H. Mustafa. Combinatorics of local search: An optimal 4-local Hall's theorem for planar graphs. In *Proc. 25th Annual European Symposium on Algorithms (ESA)*, volume 87 of *LIPIcs*, pages 8:1–8:13, 2017. `doi:10.4230/LIPIcs.ESA.2017.8`.
3. Sebastian Berndt, Valentin Dreismann, Kilian Grage, Klaus Jansen, and Ingmar Knof. Robust online algorithms for certain dynamic packing problems. In *Proc. 17th International Workshop on Approximation and Online Algorithms (WAOA)*, volume 11926 of *Lecture Notes in Computer Science*, pages 43–59, 2019. `doi:10.1007/978-3-030-39479-0_4`.
4. Sebastian Berndt, Leah Epstein, Klaus Jansen, Asaf Levin, Marten Maack, and Lars Rohwedder. Online bin covering with limited migration. In *Proc. 27th Annual European Symposium on Algorithms (ESA)*, volume 144 of *LIPIcs*, pages 18:1–18:14, 2019. `doi:10.4230/LIPIcs.ESA.2019.18`.
5. Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis*. Cambridge University Press, 1998.
6. Joan Boyar, Stephan J. Eidenbenz, Lene M. Favrholdt, Michal Kotrbcík, and Kim S. Larsen. Online dominating set. *Algorithmica*, 81(5):1938–1964, 2019. `doi:10.1007/s00453-018-0519-1`.
7. Joan Boyar, Lene M. Favrholdt, Michal Kotrbcík, and Kim S. Larsen. Relaxing the irrevocability requirement for online graph algorithms. *Algorithmica*, 84(7):1916–1951, 2022. `doi:10.1007/s00453-022-00944-w`.
8. Miroslav Chlebík and Janka Chlebíková. Approximation hardness of dominating set problems in bounded degree graphs. *Inf. Comput.*, 206(11):1264–1275, 2008. `doi:10.1016/j.ic.2008.07.003`.
9. Minati De, Sambhav Khurana, and Satyam Singh. Online dominating set and independent set. *CoRR*, abs/2111.07812, 2021. `arXiv:2111.07812`.
10. Mark de Berg, Hans L. Bodlaender, Sándor Kisfaludi-Bak, Dániel Marx, and Tom C. van der Zanden. A framework for Exponential-Time-Hypothesis-tight algorithms and lower bounds in geometric intersection graphs. *SIAM J. Comput.*, 49:1291–1331, 2020. `doi:10.1137/20M1320870`.
11. Mark de Berg, Sándor Kisfaludi-Bak, Morteza Monemizadeh, and Leonidas Theocharous. Clique-based separators for geometric intersection graphs. In *Proc. 32nd International Symposium on Algorithms and Computation (ISAAC)*, volume 212 of *LIPIcs*, pages 22:1–22:15, 2021. `doi:10.4230/LIPIcs.ISAAC.2021.22`.

12 Mark de Berg, Arpan Sadhukhan, and Frits C. R. Spieksma. Stable approximation algorithms for the dynamic broadcast range-assignment problem. In *Proc. 18th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, volume 227 of *LIPIcs*, pages 15:1–15:21, 2022. doi:10.4230/LIPIcs.SWAT.2022.15.

13 Oliver Göbel, Martin Hoefer, Thomas Kesselheim, Thomas Schleiden, and Berthold Vöcking. Online independent set beyond the worst-case: Secretaries, prophets, and periods. In *Proc. 41st International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 8573 of *Lecture Notes in Computer Science*, pages 508–519, 2014. doi:10.1007/978-3-662-43951-7_43.

14 Anupam Gupta, Ravishankar Krishnaswamy, Amit Kumar, and Debmalya Panigrahi. Online and dynamic algorithms for set cover. In *Proc. 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 537–550, 2017. doi:10.1145/3055399.3055493.

15 Anupam Gupta, Amit Kumar, and Cliff Stein. Maintaining assignments online: Matching, scheduling, and flows. In Chandra Chekuri, editor, *Proc. 2th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 468–479, 2014. doi:10.1137/1.9781611973402.35.

16 Magnús M. Halldórsson, Kazuo Iwama, Shuichi Miyazaki, and Shiro Taketomi. Online independent sets. *Theor. Comput. Sci.*, 289(2):953–962, 2002. doi:10.1016/S0304-3975(01)00411-X.

17 Sariel Har-Peled and Kent Quanrud. Approximation algorithms for polynomial-expansion and low-density graphs. In *Proc. 23rd Annual European Symposium on Algorithms (ESA)*, volume 9294 of *Lecture Notes in Computer Science*, pages 717–728. Springer, 2015. doi:10.1007/978-3-662-48350-3_60.

18 Gow-Hsing King and Wen-Guey Tzeng. On-line algorithms for the dominating set problem. *Inf. Process. Lett.*, 61(1):11–14, 1997. doi:10.1016/S0020-0190(96)00191-3.

19 Richard J. Lipton and Robert Endre Tarjan. A separator theorem for planar graphs. *SIAM J. Appl. Math*, 36(2):177–189, 1977. doi:doi/10.1137/0136016.

20 Richard J. Lipton and Andrew Tomkins. Online interval scheduling. In *Proc. 5th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 302–311, 1994. URL: http://dl.acm.org/citation.cfm?id=314464.314506.

21 Hsiang-Hsuan Liu and Jonathan Toole-Charignon. The power of amortized recourse for online graph problems. *CoRR*, abs/2206.01077, 2022. doi:10.48550/arXiv.2206.01077.

22 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proc. 38th Annual ACM Symposium on Theory of Computing (STOC)*, pages 681–690, 2006. doi:10.1145/1132516.1132612.

A Lower-bound example for Directed-DomSet

Fig. 2 shows that the Directed-DomSet has approximation ratio $\Omega(d^2)$. First, vertex a_1 arrives, which will be put into U and D_{alg}. Then a_2 and $v_1, \ldots, v_{d^2}$ arrive, with an outgoing edge to a_1. These vertices are dominated by a_1, so U and D_{alg} are not modified. Next, $w_1, \ldots, w_d$ arrive. Vertex w_1 arrives with neighbors $v_1, \ldots, v_d$. We have $U = D_{\text{alg}} = \{a_1\}$, so w_1 is not dominated by any node in D_{alg} and it is not related to any node in D_{alg}. Hence, w_1 will be put into U and D_{alg}. Then vertex w_2 arrives with neighbors $v_{d+1}, \ldots, v_{2d}$. We have $U = D_{\text{alg}} = \{a_1, w_1\}$, so w_1 is not dominated by any node in D_{alg} and it is not related to any node in D_{alg}. Hence, w_2 will be put into U and D_{alg}. Similarly, $w_3, \ldots, w_d$ will all be put into U and D_{alg}. Then $x_1, \ldots, x_{d(d-1}$ arrive, each with one outgoing edge, to a unique node among $v_1, \ldots, v_{d^2}$, as shown in the figure. They are all related to a node in U, namely one of the w_i's, so their out-neighbors will be out into D_{alg}. Finally, z arrives, which is also put into D_{alg}. Hence, at the end of the algorithm $|D_{\text{alg}}| = d^2 + 2$, but $\text{OPT} = 3$ since $\{a_1, a_2, z\}$ is a dominating set.

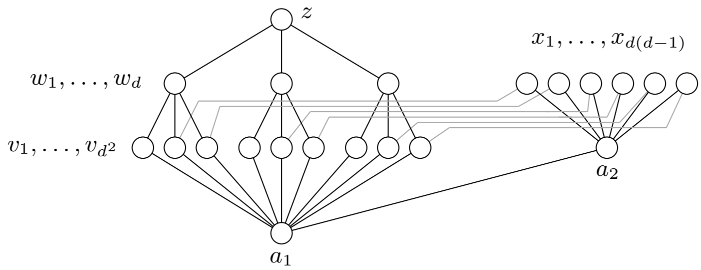

Figure 2 Example showing that DIRECTED-DOMSET has approximation ratio $\Omega(d^2)$.