Fine-Grained Complexity of Maximum Independent Sets in Circle Graphs

Nicholas Nash

October 11, 2026

Abstract

Given n chords whose $2n$ end-points are distinct points on a circle, a largest pairwise noncrossing subfamily—equivalently, a maximum independent set of the represented circle graph—can be computed in $\mathcal{O}(n^2)$ time. We prove that, under the k-Cycle Hypothesis, this problem admits no truly subquadratic algorithm. The lower bound uses a linear-size reduction from cyclically layered directed k-cycle instances. The resulting MIS optimum equals an explicit baseline if the input digraph has no directed k-cycle and exceeds that baseline by exactly one otherwise.

A known output-sensitive algorithm for maximum independent set in circle graphs runs in $\mathcal{O}(n \min\{\alpha, d\})$ time, where α is the independence number and d is the maximum number of intervals covering a gap in the supplied interval model. We give a deterministic combinatorial algorithm running in $\widetilde{\mathcal{O}}(n + \min\{\alpha^{3/2}d, \alpha d^{3/2}\})$ time. Under the Combinatorial Clique Hypothesis, its exponent is optimal among combinatorial algorithms at every feasible fixed polynomial scale where it improves on the known bound. Separately, under the fixed 5-Clique case of this hypothesis, for every $\varepsilon > 0$ we rule out combinatorial algorithms running in $\mathcal{O}(n + (\alpha d)^{5/4-\varepsilon})$ time.

1 Introduction

A circle graph is the intersection graph of chords of a circle. Equivalently, it is the overlap graph of intervals on a line [13]. Cutting a chord model at a boundary gap orders its end-points on a line. Replacing each chord by the closed interval between its end-points gives an *interval model*. Figure 1 illustrates a chord model, its interval model, and the represented graph. We defer the precise input model and the conventions used by our reductions to Section 4.

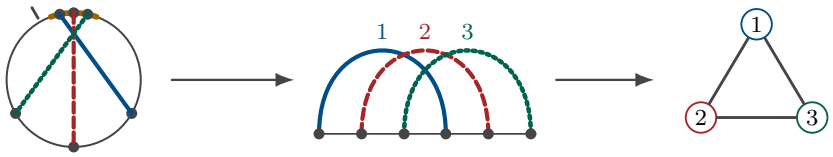


Figure 1: Three views of K_3: a chord model, the interval model obtained by cutting the circle at the marked gap, and the represented graph.

We study finding a maximum independent set (MIS) of a circle graph, that is, a largest pairwise noncrossing subfamily of the supplied chords. Applications of MIS and its weighted variant include channel routing [18], loop register allocation [12], and RNA folding and pseudoknot removal [16, 10].

Previous algorithms. Given the chord model of an n-vertex circle graph, Gavril's algorithm finds a maximum independent set in $\mathcal{O}(n^3)$ time [13]. Subsequent algorithms take $\mathcal{O}(n^2)$ time [18, 6]. For the supplied interval model, let d be the maximum number of intervals covering

a gap. Thus $d = 3$ in Figure 1, and always $d \leq n$. Apostolico et al. gave an $\mathcal{O}(nd)$-time algorithm [4, 5]. In fact, Buckingham had already given an algorithm with a running-time bound at least as good: $\mathcal{O}(n + C)$ [8], where C counts strict-containment pairs in the supplied interval model. Since each interval is contained in at most $d - 1$ others, $n + C \leq nd$. Valiente obtained the $\mathcal{O}(nd)$ bound with linear space for the weighted problem [19]. Nash and Gregg gave an $\mathcal{O}(n \min\{\alpha, d\})$-time algorithm for unweighted MIS, where α is the independence number [15].

For both unweighted and weighted MIS, the best of these bounds is quadratic in the worst case.

Related lower bounds. Bose et al. show that MIS on certain outerstring representations is at least as hard as circle-graph MIS [7]. RNA folding seeks a largest set of admissible base pairs that neither cross nor share a position, equivalently MIS on the corresponding chords. Its conditional lower bounds do not yield a near-quadratic bound for an explicit chord model: a length-L instance may have $n = \Theta(L^2)$ admissible pairs. Thus the known combinatorial[1] $L^{3-o(1)}$ barrier [1, 9] becomes only $n^{3/2-o(1)}$ after such an expansion.

Motivation. Two observations motivate our questions.

- For a uniformly random chord model—a uniformly random perfect matching of $2n$ labelled boundary positions—the independence number is $\Theta(\sqrt{n})$ with high probability [2]. Thus the deterministic Nash–Gregg algorithm runs in $\mathcal{O}(n^{3/2})$ time with high probability over the input model. Can a truly subquadratic bound also be achieved in the worst case?
- The known $\mathcal{O}(n \min\{\alpha, d\})$ bound uses only the smaller parameter. Can α and d be exploited jointly, for example in $\widetilde{\mathcal{O}}(n + \alpha d)$ time?[2]

2 Our Results

Worst-case complexity. We conditionally explain the quadratic barrier.

Theorem 1 (Conditional quadratic barrier)**.** *Under the k-Cycle Hypothesis (see Hypothesis 5), for every $\varepsilon > 0$ there is no $\mathcal{O}(n^{2-\varepsilon})$-time word-RAM algorithm for MIS on a supplied unweighted distinct-end-point chord model with n chords.*

For each fixed k, after deleting isolated vertices, our reduction maps a layered k-cycle instance with m arcs to a chord model of a circle graph with $\mathcal{O}(m)$ chords, with an exact additive-one gap.

Joint use of independence number and density. We next show that a joint $\widetilde{\mathcal{O}}(n + \alpha d)$ running-time bound is conditionally unattainable for combinatorial algorithms, even when both α and d are sublinear.

Theorem 2 (Joint-parameter lower bound)**.** *Assume the 5-Clique case of the Combinatorial Clique Hypothesis (see Hypothesis 9). For every $\varepsilon > 0$, no combinatorial word-RAM algorithm for MIS runs in $\mathcal{O}(n + (\alpha d)^{5/4-\varepsilon})$ on all supplied unweighted distinct-end-point interval models. This holds already when $\alpha, d = \Theta(n^{2/3})$.*

Nevertheless, the parameters can be exploited jointly.

Theorem 3 (Joint-parameter upper bound)**.** *A maximum independent set of a nonempty supplied unweighted distinct-end-point interval model can be computed by a deterministic combinatorial word-RAM algorithm in $\widetilde{\mathcal{O}}(n + \min\{\alpha^{3/2} d, \alpha d^{3/2}\})$ time and $\mathcal{O}(\alpha d)$ words of space.*

[1] Here and throughout, *combinatorial* excludes speedups based on fast algebraic matrix multiplication.

[2] The notation $\widetilde{\mathcal{O}}$ suppresses polylogarithmic factors.

Combining our algorithm (Section 7) with Nash and Gregg's [15] gives $\widetilde{\mathcal{O}}(n + \mathcal{T}(n, \alpha, d))$ time, where

$$\mathcal{T}(n, \alpha, d) = \min\{n\alpha, nd, \alpha^{3/2}d, \alpha d^{3/2}\}. \tag{1}$$

Theorem 4 (Conditional optimality)**.** *Assume the Combinatorial Clique Hypothesis. Fix $x, y \in (0, 1]$ with $x + y \geq 1$ and $\min\{x, y\} \leq 2/3$. For every $\varepsilon > 0$, no combinatorial word-RAM algorithm for MIS runs in $\mathcal{O}(n + \mathcal{T}(n, \alpha, d)^{1-\varepsilon})$ time on all supplied unweighted distinct-end-point interval models with $\alpha = \Theta(n^x)$ and $d = \Theta(n^y)$.*

Section 6.4 proves this theorem and shows that its range includes every feasible fixed polynomial scale where our algorithm improves. At $\alpha = d = \Theta(\sqrt{n})$, for example, the conditionally optimal bound is $\widetilde{\mathcal{O}}(n^{5/4})$, improving on $\mathcal{O}(n^{3/2})$. The theorem also covers one parameter of order n and the other of order n^c, for any fixed $c \in (0, 2/3]$.

3 Methodological disclosure

The novel ideas in this paper were generated by ChatGPT 5.6 Sol, which also assisted with drafting and revising the exposition and LaTeX. All figures were drafted and revised as TikZ code with the same system. The human authors made all final scientific and editorial decisions and have substantially simplified and independently verified all results. We take full responsibility for all submitted content.

4 Chord-model conventions

A *shared-end-point chord model* is a pair $H = (P, E)$, where P is a finite cyclically ordered set of boundary positions and $E \subseteq \binom{P}{2}$ is a set of chords. We assume that every position is used. Two chords *conflict* if they share a position or, when their four end-points are distinct, the end-points alternate in the cyclic order. The *conflict graph* $G(H)$ has vertex set E and joins exactly the conflicting pairs. The model has *distinct end-points* if no two chords share a position. An *independent chord family* is a set of pairwise nonconflicting chords.

The shared- and distinct-end-point models are equivalent under a simple perturbation that preserves the conflict graph. Specifically, if q chords have an end-point at one boundary position, replace it by q consecutive distinct positions, one for each chord, in the cyclic order of their other end-points. The formerly incident chords then pairwise cross, while no other conflict changes. Applying this replacement at every shared position gives a distinct-end-point chord model with the same conflict graph and number of chords [13].

Write $n = |E|$ and $p = |P|$. An explicit instance specifies the cyclic order as $1, \ldots, p$ and lists the n end-point pairs, with weights when relevant. Since all positions are used, this requires $\mathcal{O}(n)$ words. Unless stated otherwise, identifiers are polynomially bounded integers, the word size is $\Theta(\log n)$, and running times are deterministic worst-case bounds. By computing MIS we mean returning a maximum independent chord family (and hence its cardinality $\alpha(G(H))$). In a weighted model, $w(M)$ is the total weight of an independent chord family M and $\mathrm{MWIS}(H)$ is the maximum such value. The notation α continues to denote the unweighted independence number.

For the parameter-sensitive statements, the input also specifies a boundary gap at which to cut the chord model, giving the supplied interval model. We use the same label for a chord and its interval. Two intervals are compatible exactly when they are disjoint or one lies strictly inside the other. An interval covers a gap when one end-point lies before that gap and the other after it. The density d of the supplied interval model is the maximum number of intervals covering one gap. The known parameter-sensitive algorithms use this d for any interval model obtained from the supplied chord model.

5 From directed cycles to circle graph MIS

We use the following form of the k-Cycle Hypothesis [11].

Hypothesis 5 (k-Cycle Hypothesis)**.** *For every $\varepsilon > 0$, there is a constant k for which no $\mathcal{O}(m^{2-\varepsilon})$-time word-RAM algorithm detects a directed k-cycle in an m-edge digraph.*

A *cyclically layered digraph* has disjoint vertex layers $V_0, \ldots, V_{k-1}$, where $k \geq 3$, and arcs only in the *interfaces* $A_i \subseteq V_i \times V_{(i+1) \bmod k}$. A *one-lap cycle* uses exactly one arc from each interface. These are precisely its directed k-cycles. The deterministic k-circle-layering reduction [14, Lemma 2.2], using colour coding [3, Section 4], turns a digraph with N vertices and m arcs into at most $k^{\mathcal{O}(k)} \log N$ such layered subgraphs, each with at most m arcs. The original graph has a directed k-cycle exactly when one subgraph does, and the total construction time is $k^{\mathcal{O}(k)}(N+m)\log N$.

The chord construction used by our reduction needs a perfect matching as its last interface. To arrange this, add a disjoint copy $V_k = \{v^+ : v \in V_0\}$, replace each arc $(u, v) \in A_{k-1}$ by (u, v^+), and add every arc (v^+, v). Call the resulting digraph the *identity extension* D^+. It has $N + r$ vertices and $m + r$ arcs, where $r = |V_0|$, and takes $\mathcal{O}(N + m)$ time to construct. Contracting the new matching gives a bijection between its one-lap cycles and those of D.

5.1 The chord construction and its exact gap

We state the construction for any layered digraph with a closing perfect matching. Section 6.1 re-uses this construction but will instead provide such digraphs directly.

Definition 6 (The chord model K_D)**.** Let D have layers $V_0, \ldots, V_{k-1}$, N vertices, and M arcs, with a perfect matching $A^\star = A_{k-1}$ from V_{k-1} to V_0. For each non-closing interface A_i, a boundary section S_i lists the *source pairs* $\langle Y_u, R_u \rangle$ for $u \in V_i$ and *target pairs* $\langle L_v, X_v \rangle$ for $v \in V_{i+1}$. Each pair is consecutive, each occurs once, and the source pair of every arc precedes its target pair.

Order the matching as $A^\star = \{(u_1, v_1), \ldots, (u_t, v_t)\}$ and set

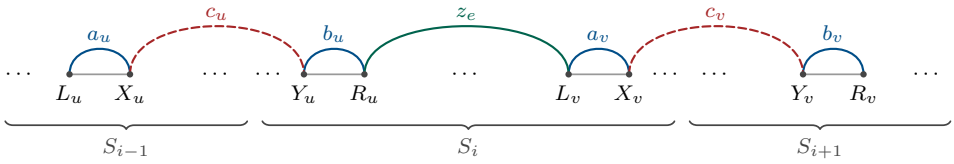
$$S^\star = \langle Y_{u_1}, R_{u_1}, \ldots, Y_{u_t}, R_{u_t}, L_{v_1}, X_{v_1}, \ldots, L_{v_t}, X_{v_t} \rangle.$$

The cyclic boundary order is $S_0 \circ \cdots \circ S_{k-2} \circ S^\star$, where $\circ$ denotes concatenation. Write $S_{k-1} = S^\star$ and read section indices modulo k. For each arc $e = (u, v)$, add the arc chord $z_e = R_u L_v$. For each vertex v, also add the chords

$$a_v = L_v X_v, \quad b_v = Y_v R_v, \quad c_v = X_v Y_v.$$

Give each closing arc chord z_e, $e \in A^\star$, weight two, and all other chords weight one. The model has $4N$ boundary positions and $M + 3N$ chords.

For a non-closing arc $e = (u, v) \in A_i$, the local order is illustrated below.



Lemma 7 (Exact additive-one gap)**.** *The model K_D satisfies* $\mathrm{MWIS}(K_D) = 2N + 1$ *if D contains a one-lap cycle, and* $\mathrm{MWIS}(K_D) = 2N$ *otherwise.*

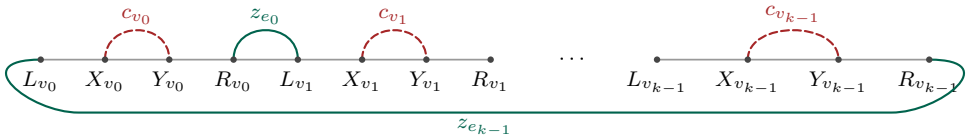
Proof. The chords a_v, b_v form an independent family of weight $2N$, since they join disjoint consecutive pairs of boundary positions. Let I be any independent family, let T be its selected arcs (that is, $e \in T$ exactly when $z_e \in I$), and let $V(T)$ be their incident vertices. Two selected arcs cannot share a source or a target: their chords would share R_v or L_v. Thus the digraph $(V(T), T)$ is a disjoint union of directed paths and cycles. If $\pi(T)$ counts its path components, then $|V(T)| - |T| = \pi(T)$.

At a vertex incident with T, an arc chord excludes either a_v or b_v, and the other two local chords conflict. This vertex contributes at most one local chord, rather than two. The closing arc chords pairwise cross, because their source and target end-points have the same order. Hence $b^\star := |T \cap A^\star| \leq 1$, and

$$w(I) \leq 2N - |V(T)| + |T| + b^\star = 2N - \pi(T) + b^\star \leq 2N + 1. \tag{2}$$

Exceeding $2N$ requires $\pi(T) = 0$ and $b^\star = 1$. Every directed cycle traverses the closing interface once per lap, so T must then consist of exactly one one-lap cycle.

Conversely, let $e_i = (v_i, v_{i+1})$, with indices modulo k, be a one-lap cycle. Select its arc chords, c_{v_i} at its vertices, and a_v, b_v elsewhere. The cyclic order restricted to the selected arc and c-chord end-points is as follows.



Every selected arc or c-chord joins consecutive positions in this restricted order, including $z_{e_{k-1}} = R_{v_{k-1}} L_{v_0}$ at the wraparound. Each off-cycle a_v, b_v joins consecutive positions in the full order and shares no end-point with the selected chords. The family is therefore independent and has weight $2(N-k) + k + (k+1) = 2N + 1$, attaining (2). □

5.2 Completing the reduction

Lemma 8 (Removing shared end-points and weights)**.** *The model K_D can be transformed in $\mathcal{O}(M+N)$ time into an unweighted distinct-end-point chord model J_D with $\alpha(J_D) = \mathrm{MWIS}(K_D)$ and $|E(J_D)| = M + 3N + |A^\star|$.*

Proof. Apply the shared-to-distinct perturbation of Section 4. Replace each weight-two chord c by compatible copies c_1, c_2, ordered as c_1, c_2 at one end and c_2, c_1 at the other. Every other chord conflicts with both or neither, so a maximum independent family contains both copies or neither. This preserves the optimum and adds one chord per closing arc. The end-point records can be bucket sorted in linear time. Comparison sorting instead gives $\mathcal{O}((M+N)\log(M+N))$ time. □

Proof of Theorem 1. Let D be a cyclically layered digraph with k nonempty layers, N vertices and m arcs, and put $r = |V_0|$. Applying Lemmas 7 and 8 to its identity extension D^+ gives, in $\mathcal{O}(N+m)$ time, an unweighted distinct-end-point chord model with $n = (m+r)+3(N+r)+r = m+3N+5r$ chords. Its independence number is $2(N+r)+1$ if D contains a directed k-cycle, and $2(N+r)$ otherwise. If D has no isolated vertices, then $N \leq 2m$ and $r \leq m$, so $n \leq 12m$.

Suppose MIS takes $\mathcal{O}(n^{2-\varepsilon})$ time. We may assume $0 < \varepsilon < 1$. Choose k from Hypothesis 5 with improvement $\varepsilon/2$. Delete isolated vertices from a directed k-Cycle instance, so $N \leq 2m$, and apply the deterministic layering reduction above. Delete isolated vertices from each layered instance and discard it if a layer becomes empty. The preceding construction gives at most $12m$ chords for each remaining instance. The assumed MIS algorithm would decide the original instance, including its colour-coding construction, in

$$k^{\mathcal{O}(k)} \log N \cdot \mathcal{O}(m + m^{2-\varepsilon}) = \mathcal{O}(m^{2-\varepsilon/2}),$$

contradicting the choice of k. □

6 Parameter-sensitive lower bounds

We prove Theorems 2 and 4 using the following hypothesis.

Hypothesis 9 (Combinatorial Clique Hypothesis)**.** *For every fixed integer $K \geq 3$ and every $\delta > 0$, no combinatorial word-RAM algorithm detects a K-clique in a K-partite graph with Q vertices per part in $\mathcal{O}(Q^{K-\delta})$ time.*

6.1 The six-layer construction

The following is the pair case of the Clique-to-Cycle construction [14, Theorem 3.1 and Lemma 4.1], followed by the identity extension of Section 5.

Definition 10 (Six-layer digraph)**.** Let F be a 5-partite graph with ordered parts $C_0, \ldots, C_4$, each of size $Q \geq 1$. Its digraph D_F has disjoint layers $W_i = C_i \times C_{(i+1) \bmod 5}$ for $0 \leq i < 5$, and a second copy W_5 of W_0. For $0 \leq i < 5$, include the arc

$$(x, y) \longrightarrow (y, z) \quad \text{from } W_i \text{ to } W_{i+1} \quad \Longleftrightarrow \quad \{x, y, z\} \text{ is a clique in } F.$$

The closing interface is the identity matching from W_5 to W_0.

Lemma 11 (Correctness of the six-layer digraph)**.** *The graph F has a 5-clique if and only if D_F has a one-lap cycle.*

Proof. By the arc definition and closing matching, the vertices of a one-lap cycle, in layer order, have the form

$$(v_0, v_1),\ (v_1, v_2),\ (v_2, v_3),\ (v_3, v_4),\ (v_4, v_0),\ (v_0, v_1),$$

where $v_i \in C_i$. The first and last vertices lie in W_0 and W_5. The five non-closing arcs require each triple $\{v_i, v_{i+1}, v_{i+2}\}$ to be a clique, with subscripts modulo five. Every pair of the five vertices occurs in one of these triples, so $\{v_0, \ldots, v_4\}$ is a clique in F. Conversely, a 5-clique supplies such a one-lap cycle. □

We now apply Definition 6 to D_F, choosing its boundary order to control the density of the resulting chord (or interval) model. For each non-closing section S_i, $0 \leq i < 5$, and each $y \in C_{(i+1) \bmod 5}$, list all source pairs $\langle Y_{(x,y)}, R_{(x,y)} \rangle$ for $x \in C_i$, then all target pairs $\langle L_{(y,z)}, X_{(y,z)} \rangle$ for $z \in C_{(i+2) \bmod 5}$. Call this consecutive list of boundary positions the *block* $B_i(y)$, and concatenate the blocks to form S_i. Every non-closing arc chord has both end-points in one block, with its source pair before its target pair, as required by Definition 6. Each block has Q source pairs, Q target pairs, and at most Q^2 arc chords. Figure 2 illustrates one block.

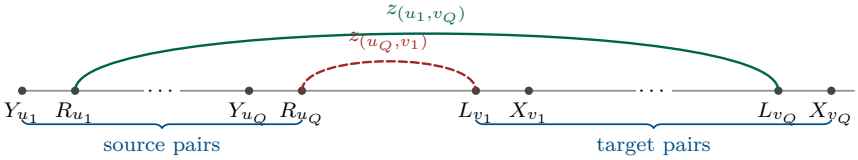


Figure 2: One boundary block $B_i(y)$, with $u_j = (x_j, y)$ and $v_j = (y, z_j)$ listing its source and target vertices. Two possible arc chords are shown. All arc chords for this value of y have both end-points in this block. Different values of y occupy disjoint consecutive blocks.

Lemma 12 (Size and padding)**.** *From the six-layer digraph D_F of Definition 10, one can construct in $\mathcal{O}(Q^3)$ time an unweighted distinct-end-point interval model with $n = \Theta(Q^3), d = \Theta(Q^2)$. Its independence number is $B_Q = 12Q^2 + Q$ when F has no 5-clique, and $B_Q + 1$ otherwise. The minimum density over all shared- or all distinct-end-point interval models of its graph is also $\Theta(Q^2)$.*

Proof. The digraph D_F has $N = 6Q^2$ vertices, at most $5Q^3 + Q^2$ arcs, and Q^2 closing arcs. Lemmas 7, 8, and 11 give $\mathcal{O}(Q^3)$ chords and independence number $12Q^2$ or $12Q^2 + 1$.

To bound density, first cut the weighted model K_{D_F} just before S_0. The blocks $B_i(y)$ are disjoint consecutive lists of positions, and each non-closing arc chord has both end-points in one such list. Thus at most Q^2 non-closing arc intervals cover a gap. The Q^2 closing intervals, $6Q^2$ intervals c_v, and at most one a_v or b_v add $\mathcal{O}(Q^2)$, giving density $\mathcal{O}(Q^2)$. At most $Q + 1$ chords meet a shared position. Each new gap created by separating that position is covered by at most $Q + 1$ intervals in addition to those at an adjacent old gap. Density therefore remains $\mathcal{O}(Q^2)$. Duplicating closing chords adds at most Q^2.

One copy of each closing chord forms a clique of order Q^2. In any interval model of the resulting graph, these Q^2 intervals pairwise intersect and hence have a common point. With distinct end-points, their intersection contains a gap, giving density at least Q^2. With shared end-points, at least half extend to the same side of the common point and cover a common gap, giving density at least $Q^2/2$. Thus every representation has density $\Omega(Q^2)$.

Finally, add Q clique components of order Q^2, each in its own portion of the line after the existing intervals. They add Q^3 intervals, giving $n = \Theta(Q^3)$, and exactly Q to the independence number, while density remains $\Theta(Q^2)$. Constructing D_F from F, generating the end-point lists, and the transformations take $\mathcal{O}(Q^3)$ time. □

Proof of Theorem 2. By Lemma 12, $n = \Theta(Q^3)$ and $\alpha d = \Theta(Q^4)$. The excluded algorithm would decide 5-Clique, including construction, in $\mathcal{O}(Q^3 + Q^{5-4\varepsilon}) = \mathcal{O}(Q^{5-\delta})$ time for $\delta = \min\{2, 4\varepsilon\} > 0$. This contradicts the $K = 5$ case of Hypothesis 9. The same instances rule out $\mathcal{O}(n^{5/3-\varepsilon})$ time, and hence a universal $\widetilde{\mathcal{O}}(n + \alpha d)$ bound. □

6.2 Combining several digraphs in one chord model

To prove conditional optimality (Theorem 4), we combine several six-layer digraphs. Sharing their reference chords keeps α of order Q^2 while d grows. Placing their interval models consecutively instead adds their independence numbers but takes only their maximum density. This lets us vary the two parameters separately.

Let F now be $(G + 5)$-partite, with parts $C_0, \ldots, C_4$ and G additional parts $P_1, \ldots, P_G$, each of size Q, where $G \geq 0$ is fixed. For each tuple $p \in P_1 \times \cdots \times P_G$, choosing one vertex from each P_i, let D_p be the digraph on the six layers $W_0, \ldots, W_5$ of Definition 10. Its non-closing arcs are $(x, y) \to (y, z)$ exactly when x, y, z together with the entries of p form a clique, and its closing interface is the identity matching. The same argument as in the proof of Lemma 11 gives

$$D_p \text{ has a one-lap cycle} \quad \Longleftrightarrow \quad p \text{ extends to a } (G + 5)\text{-clique in } F.$$

Select $C \geq 1$ distinct tuples $p_1, \ldots, p_C$ and write $D_\gamma = D_{p_\gamma}$. We seek a single chord model whose optimum determines whether at least one D_γ has a one-lap cycle. Applying Definition 6 to the digraph with arc set $A(D_1) \cup \cdots \cup A(D_C)$ is insufficient: a one-lap cycle in this union need not belong to any individual D_γ. The construction below assigns colour γ to each arc of D_γ to preserve this distinction.

Definition 13 (Chord model for several layered digraphs)**.** Let $D_1, \ldots, D_C$, where $C \geq 1$, be cyclically layered digraphs with common nonempty layers $V_0, \ldots, V_{k-1}$, $k \geq 3$, and a common ordered closing perfect matching. Represent each arc of D_γ with colour γ, separately for each γ. For each non-closing interface i, let $P^-_{i,1}, \ldots, P^-_{i,t_i}$ partition V_i and $P^+_{i,1}, \ldots, P^+_{i,t_i}$ partition V_{i+1}. Every arc $u \to v$ in this interface, in every D_γ, must satisfy $u \in P^-_{i,j}$ and $v \in P^+_{i,j}$ for some j.

In the section S_i of Definition 6, let the *boundary block* $B_{i,j}$ list all source pairs for $P^-_{i,j}$, then all target pairs for $P^+_{i,j}$. Concatenate these lists in order of j, and retain the closing section $S^\star$.

Replace each source pair and each target pair by the respective lists

$$Y_v, y_{v,1}, \rho_{v,1}, \ldots, y_{v,C}, \rho_{v,C}, R_v,$$
$$L_v, \lambda_{v,1}, x_{v,1}, \ldots, \lambda_{v,C}, x_{v,C}, X_v.$$

Call these lists the *source segment* and *target segment* of v. Retain the unit-weight *reference chords* $a_v = L_v X_v$ and $b_v = Y_v R_v$. Replace c_v by unit-weight *connector chords* $c_{v,\tau} = x_{v,\tau} y_{v,\tau}$, one for each colour τ. For each arc $u \to v$ of colour γ, add $z_{\gamma,u,v} = \rho_{u,\gamma}\lambda_{v,\gamma}$, of weight two for a closing arc and one otherwise. Delete unused boundary positions. Denote the resulting weighted chord model by K.

Write $N = \sum_i |V_i|$, $M = \sum_\gamma |A(D_\gamma)|$, $\lambda = \max_i |V_i|$, and let b be the largest number of arc chords of one colour with both end-points in one non-closing block $B_{i,j}$.

Theorem 14 (Sharing the reference chords)**.** *For the model K of Definition 13,* $\mathrm{MWIS}(K) = 2N$ *if no D_γ has a one-lap cycle, and $2N+1$ otherwise. In $\mathcal{O}(M + CN)$ time it gives an unweighted distinct-end-point interval model with the same optimum, $\mathcal{O}(M + CN)$ intervals, and density $\mathcal{O}(C(b+\lambda))$. The resulting graph contains a clique of order $C|V_0|$.*

Proof. The reference chords give $\mathrm{MWIS}(K) \geq 2N$. To prove the upper bounds, let I be any independent family and count how many reference chords its other chords exclude.

Represent the target and source segments of each v by auxiliary vertices: a *target vertex* v^- and a *source vertex* v^+. Each selected connector gives a *connector edge* $\{v^-, v^+\}$. Each selected arc chord $z_{\gamma,u,v}$ gives an *arc edge* $\{u^+, v^-\}$. Let T be these edges, $V(T)$ their incident vertices, and $H = (V(T), T)$. No two selected chords give the same edge: connectors for one vertex pairwise cross, as do different-colour chords for the same arc. An arc edge cannot equal a connector edge, since arcs join different layers. Thus H is a simple undirected graph.

Every selected arc or connector chord joins two distinct boundary segments and crosses the reference chord spanning each of them. Consequently, I contains at most $2N - |V(T)|$ reference chords. The closing arc chords pairwise cross, since their source and target end-points have the same order, first by matching arc and then by colour. Their number $b^\star$ in I is therefore zero or one, giving

$$w(I) \leq 2N - |V(T)| + |T| + b^\star. \tag{3}$$

If T is empty this is at most $2N$. If H is a nonempty forest, $|T| \leq |V(T)| - 1$ gives the same bound. Thus exceeding $2N$ requires a cycle in H.

We show that cycles of H alternate between arc and connector edges. Draw H with straight edges and its vertices on a circle in the order of their boundary segments. Its edges cannot cross, since the corresponding chords in I would cross.

Suppose a cycle uses arc edges $\{w, a\}$ and $\{w, b\}$. The boundary arc from a to b avoiding w lies within one consecutive source or target portion of $B_{i,j}$ (or $S^\star$ for closing arcs). All vertices of the remaining a-to-b path must lie there: a vertex outside would be separated from a by $\{w, b\}$, or from b by $\{w, a\}$, forcing a crossing. But no two vertices in that portion are adjacent in H (Figure 3), a contradiction.

Each auxiliary vertex has at most one connector edge. The two edges used by a cycle at each vertex must therefore be one arc edge and one connector edge, so they alternate.

Follow an arc edge from u^+ to v^- around a cycle of H. The next edge must be the connector $\{v^-, v^+\}$, followed by an arc edge representing an arc leaving v. The represented arcs therefore form a directed cycle through the original layers. It must traverse the closing interface, and does so only once since $b^\star \leq 1$. Thus it makes one lap.

At each vertex v on this directed cycle, let γ, τ, γ' be the incoming arc, connector, and outgoing arc colours. To avoid crossing the connector, $\lambda_{v,\gamma}$ must precede $x_{v,\tau}$, and $y_{v,\tau}$ must precede $\rho_{v,\gamma'}$. The two end-point lists of Definition 13 give, respectively,

$$\gamma \leq \tau \leq \gamma'. \tag{4}$$

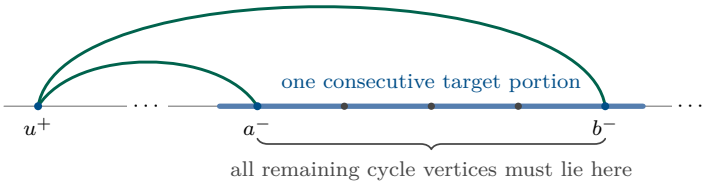


Figure 3: A cycle cannot use both drawn arc edges at u^+. Its remaining vertices would all lie between a^- and b^-, in a portion containing only target vertices. No auxiliary edge joins two such vertices.

Figure 4 illustrates the first inequality. The arc colours cannot decrease around a closed cycle, so all arc and connector colours are equal. The cycle belongs to one D_γ.

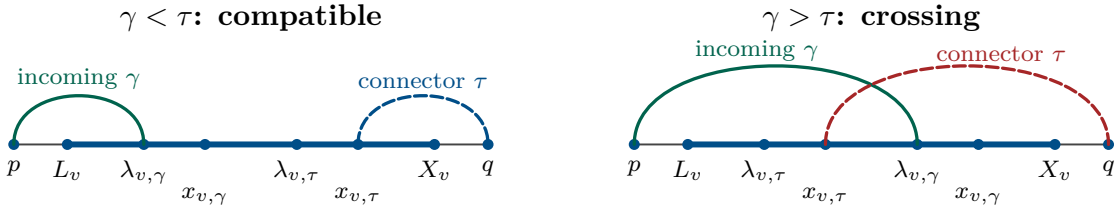


Figure 4: At the target segment of v, an incoming arc chord of colour γ is compatible with a connector of colour τ exactly when $\gamma \leq \tau$. Equality is compatible because $\lambda_{v,\gamma}$ precedes $x_{v,\gamma}$. Here p, q are the other end-points.

Deleting from H its edge representing a closing arc, if present, breaks every cycle. Thus $|T| \leq |V(T)|$, and (3) gives $w(I) \leq 2N + 1$. Since I was arbitrary, $\mathrm{MWIS}(K) \leq 2N + 1$. If no D_γ has a one-lap cycle, H is itself a forest, giving $w(I) \leq 2N$. The reference family then shows that $\mathrm{MWIS}(K) = 2N$.

It remains to attain weight $2N + 1$ whenever some D_γ has a one-lap cycle. Let $v_0 \to \cdots \to v_{k-1} \to v_0$ be such a cycle. Select its arc chords, its vertices' connectors of colour γ, and both reference chords at every other vertex. Among the selected arc and connector end-points, section S_i has order $y_{v_i,\gamma}, \rho_{v_i,\gamma}, \lambda_{v_{i+1},\gamma}, x_{v_{i+1},\gamma}$, with indices modulo k. Each selected arc joins consecutive positions within a section, and each selected connector does so between successive sections. The other reference segments contain no such end-point. The family is therefore independent, with weight $2(N - k) + k + (k + 1) = 2N + 1$.

Size and density. The closing matching has $|V_0|$ arcs, so the transformations of Lemma 8 give $2N + CN + M + C|V_0| = \mathcal{O}(M + CN)$ unweighted intervals with the same optimum. One copy of each closing chord forms the claimed clique.

Cut K just before S_0. Each non-closing arc interval lies within one block $B_{i,j}$, so at most Cb cover a gap. For $i > 0$, connectors for V_i join S_{i-1} to S_i. Any gap is therefore covered by connectors for at most two such layers, plus those for V_0, which join S_0 to $S^\star$. These contribute at most $3C\lambda$. Closing intervals contribute at most $C\lambda$, and reference intervals at most one.

At most $\max\{b, 1\}$ chords meet a shared position. Each new gap created by separating it has at most this many additional intervals compared with an adjacent old gap. Duplicating closing chords adds at most $C|V_0| \leq C\lambda$, giving the claimed density bound. Counting sort on the $\mathcal{O}(CN)$ boundary ranks orders the chords at each shared position by their other end-points in $\mathcal{O}(M + CN)$ time. The remaining construction steps take linear time. □

6.3 Two ways to set the parameters

Return to the six-layer digraphs $D_1, \ldots, D_C$ of Section 6.2, each with $N = 6Q^2$ vertices. We construct two interval models whose independence numbers determine whether any D_γ has a one-lap cycle.

First, combine the digraphs using Definition 13 and apply Theorem 14 to remove weights and shared end-points. The reference chords are shared, with total weight $2N$. The resulting model has independence number $12Q^2$ if no D_γ has a one-lap cycle, and $12Q^2 + 1$ otherwise.

Alternatively, apply Definition 6 and Lemma 8 to each D_γ separately, and place the resulting interval models consecutively on the line. Intervals from different models do not conflict, so their independence numbers add, giving $2CN + t = 12CQ^2 + t$. Here t counts the digraphs having a one-lap cycle, equivalently the selected tuples extending to a $(G+5)$-clique. Before adding any further intervals, the parameters are

construction	n_0	α_0	d_0
one model for all D_γ	$\mathcal{O}(CQ^3)$	$\Theta(Q^2)$	$\Theta(CQ^2)$
separate models, placed consecutively	$\mathcal{O}(CQ^3)$	$\Theta(CQ^2)$	$\Theta(Q^2)$.

(5)

Both take $\mathcal{O}(CQ^3)$ time. Here $\lambda = Q^2$ and $b \le Q^2$. Theorem 14 and the density count in Lemma 12 give the upper bounds, and the closing cliques give the lower bounds. Consecutive models add their independence numbers and take their maximum density.

The following addition of intervals lets us reach prescribed parameter sizes without changing the answer to the clique question.

Lemma 15 (Adding intervals to set the parameters)**.** *Fix $x, y \in (0, 1]$ with $x + y \ge 1$. A supplied unweighted distinct-end-point interval model with $n_0 = \mathcal{O}(P)$, $\alpha_0 = \mathcal{O}(P^x)$, and $d_0 = \mathcal{O}(P^y)$, where $P \ge 1$, can be enlarged in $\mathcal{O}(P)$ time to satisfy $n = \Theta(P)$, $\alpha = \Theta(P^x)$, and $d = \Theta(P^y)$. The increase in independence number is known explicitly.*

Proof. After the existing model, add $\lceil P^x \rceil$ isolated intervals and $J = \lceil P / \lceil P^y \rceil \rceil$ clique components, each of order $\lceil P^y \rceil$, in separate portions of the line. They add $\Theta(P)$ intervals and exactly $\Delta = \lceil P^x \rceil + J$ to the independence number. Since $J = \mathcal{O}(P^{1-y}) = \mathcal{O}(P^x)$, this gives the required bounds. The supplied density becomes $\max\{d_0, \lceil P^y \rceil\}$. A new clique also forces density $\Omega(P^y)$ in every representation. □

6.4 Conditional optimality at polynomial scales

Put $\mu = \min\{x, y\}$ and $\nu = \max\{x, y\}$.

Proof of Theorem 4. Fix the constants x, y and $\varepsilon > 0$ as in the theorem, and suppose the stated MIS algorithm exists. Put $f = \min\{1 + \mu, \nu + 3\mu/2\}$, so $\mathcal{T}(n, \alpha, d) = \Theta(n^f)$ at these scales.

Fix an integer $G \ge 2/\mu$. The source $(G+5)$-Clique instance is a $(G+5)$-partite graph F with parts $C_0, \ldots, C_4, P_1, \ldots, P_G$, each of size Q. Its Q^G choices $p \in P_1 \times \cdots \times P_G$ define the digraphs D_p of Section 6.2. For any group of C choices, (5) gives $\mathcal{O}(CQ^3)$ intervals with parameter orders Q^2 and CQ^2. Use the first construction if $x \le y$, and the second otherwise, so that C multiplies the larger target parameter.

We seek a final model with $n = \Theta(P)$, $\alpha = \Theta(P^x)$, and $d = \Theta(P^y)$. Set $P = Q^{2/\mu}$ to make $Q^2 = P^\mu$. To apply Lemma 15, the number of intervals requires $CQ^3 = \mathcal{O}(P)$, and the larger parameter requires $CQ^2 = \mathcal{O}(P^\nu)$. Thus choose

$$C = \left\lceil \min\left\{ \frac{P}{Q^3}, \frac{P^\nu}{Q^2} \right\} \right\rceil .$$

Both quantities inside the minimum are at least one, since $\mu \le 2/3$ and $\nu \ge \mu$, so rounding changes these bounds by at most a factor of two.

Partition $P_1 \times \cdots \times P_G$ into groups of C choices, giving $\mathcal{O}(Q^G/C)$ groups since $C \le Q^G$. If the last group is smaller, add digraphs on the same six layers containing only the closing matching until there are C digraphs. The added digraphs have no one-lap cycle. Construct and enlarge one interval model per group in $\mathcal{O}(P)$ time using Lemma 15. If the added intervals increase α by Δ, compare the optimum with $12Q^2 + \Delta$ for the first construction, or $12CQ^2 + \Delta$ for the second. By Section 6.3, F has a $(G+5)$-clique exactly when at least one optimum exceeds its comparison value.

Our choice of C gives

$$CQ^5 = \Theta\big(\min\{PQ^2, P^\nu Q^3\}\big) = \Theta(P^f).$$

Thus an algorithm with the excluded running time would decide whether F has a $(G+5)$-clique in total time

$$\mathcal{O}\left(\frac{Q^G}{C}\left(P + P^{f(1-\varepsilon)}\right)\right) = \mathcal{O}\left(Q^{G+5}\left(P^{-(f-1)} + P^{-f\varepsilon}\right)\right).$$

The first term includes construction. The second comes from $\mathcal{T}^{1-\varepsilon}$. Since $\mu + \nu \ge 1$, we have $f \ge 1 + \mu/2 > 1$. Both terms therefore save a fixed positive power of P, and hence of Q, contradicting the $(G+5)$-Clique hypothesis. □

The new algorithm improves on the Nash–Gregg exponent exactly when $\nu + \mu/2 < 1$. Since $\nu \ge \mu$, this implies $\mu < 2/3$. Thus the six-layer argument covers every fixed polynomial scale where the new algorithm improves.

7 The joint-parameter algorithm

We prove Theorem 3. Throughout this section, the supplied interval model is nonempty and unweighted, with distinct end-points.

7.1 Dynamic programming subproblems

Number the end-points $0, \dots, 2n-1$ and write interval j as $[\ell_j, r_j]$. For a family $\mathcal{S}$ of intervals, let $\alpha(\mathcal{S})$ be the size of a largest compatible subfamily. Recall from Section 4 that two intervals are compatible if they are disjoint or one contains the other. An interval i is open after position t if $\ell_i \le t < r_i$. Its subproblem asks for a largest compatible family of intervals strictly inside i that have closed by t. That is, define

$$F_i(t) := \alpha\big(\{h : \ell_i < \ell_h < r_h \le t\}\big).$$

When j closes, the best family consisting of j and intervals strictly inside it has size $c_j = 1 + F_j(r_j - 1)$. For every remaining open interval i with $\ell_i < \ell_j$,

$$F_i(r_j) = \max\{F_i(r_j - 1),\ F_i(\ell_j - 1) + c_j\}. \tag{6}$$

This is the standard dynamic programming recurrence from [15, Section 3], applied at a right end-point. The first term omits j. If j is included, every other chosen interval closes before ℓ_j or lies strictly inside j: other intervals either conflict with j or have not yet closed. These two parts can be optimised independently. Values for intervals with $\ell_i > \ell_j$ are unchanged. An auxiliary interval $\star = [-1, 2n]$ represents the subproblem on the whole input. It stays open throughout the sweep, and $F_\star(2n-1) = \alpha$.

We will record only the positions at which a subproblem value increases. The next lemma bounds the size and total number of these changes.

Lemma 16 (Changes in the subproblem optima)**.** *Put $U = \sum_j c_j$. Then $n \le U \le \alpha d$. For each interval i, including $\star$, we have $F_i(t) - F_i(t-1) \in \{0, 1\}$ whenever $\ell_i < t < r_i$. Summed over all such i, t, these differences total $U - n + \alpha \le U$.*

Proof. The family defining $F_i(t)$ contains that defining $F_i(t-1)$ and at most one additional interval, the one ending at t. Hence each difference is zero or one. Summing over t gives $F_i(r_i - 1) - F_i(\ell_i) = c_i - 1$ for each input interval i, and α for $\star$. The total is therefore $U - n + \alpha \le U$.

To bound U, partition the input intervals into at most d families of pairwise disjoint intervals, assigning each opening interval to a family with none open. Such a family exists because at most $d-1$ earlier intervals are open. Within one family, choose for each j an independent set of size c_j consisting of j and intervals strictly inside it. These sets occupy disjoint regions, so their union is independent and their total size is at most α. Summing over the families gives $U \le \alpha d$. Also $U \ge n$ since $c_j \ge 1$. □

7.2 Maintaining the subproblem values

Algorithm 1 processes all $2n$ end-points from left to right. For each open interval i, including $\star$, a data structure D stores a growing array f_i, initially (ℓ_i). Put $k_i = |f_i| - 1$. For $1 \le q \le k_i$, $f_i[q]$ records the first position where F_i reaches q, with $f_i[0] = \ell_i$ by convention. We call the entries *attainment positions*. The value k_i is the current subproblem value.

Suppose the arrays are correct just before j closes. For an interval i that remains open with $\ell_i < \ell_j$, the recurrence says that F_i increases exactly when $F_i(\ell_j - 1) + c_j > k_i$, or equivalently when it has reached $q = k_i - c_j + 1$ before ℓ_j. The intervals strictly inside j are already available to i, so $k_i \ge c_j - 1$. Since $c_j \ge 1$, we have $0 \le q \le k_i$, and the test is simply $f_i[q] < \ell_j$. The convention $f_i[0] = \ell_i$ covers $q = 0$. An interval starting after ℓ_j cannot pass this comparison, because every entry of its array is at least its own starting position. Thus the intervals whose subproblem values increase when j closes are exactly

$$\{i \text{ open} : 1 \le c_j \le k_i + 1 \quad \text{and} \quad f_i[(k_i - c_j) + 1] < \ell_j\}. \tag{7}$$

We maintain the arrays using four operations of D:

- Open(i) opens interval i and sets $f_i \leftarrow (\ell_i)$.
- Increase(i, t) appends t to f_i.
- Close(i) closes interval i and returns k_i.
- Query(j) is called after j closes and c_j has been computed. It returns each interval in (7) once, without changing any array.

By Lemma 16, each returned interval's value increases by one, so its array receives the single new entry r_j. Line 10 finishes constructing the result list before any values are updated. Figure 5 illustrates the test: with $k_i = 5$ and $c_j = 3$, having reached $q = 3$ before ℓ_j suffices to increase k_i to 6.

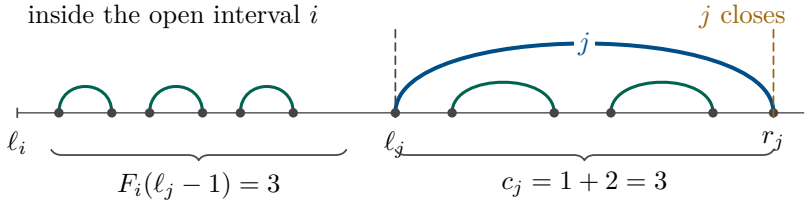


Figure 5: The five green intervals have already closed. Selecting j with the three intervals before it and the two inside it increases the subproblem value from $k_i = 5$ to 6. The outer subproblem i remains open.

Algorithm 1 Computing the independence number in one pass.

Input: An interval model $\mathcal{I}$ with distinct end-points supplied in order.
Output: The independence number of $\mathcal{I}$.

```
1: procedure IndependenceNumber(I)
2:   Initialise empty D
3:   ℓ⋆ ← −1, Open(⋆)
4:   for t = 0, ..., 2n − 1 do
5:     j ← the interval whose end-point is at position t
6:     if t = ℓ_j then
7:       Open(j)
8:     else
9:       c_j ← 1 + Close(j)
10:      T ← Query(j)
11:      for each i in T do
12:        Increase(i, r_j)
13:  return Close(⋆)
```

7.3 Correctness

Lemma 17 (Correctness of Algorithm 1)**.** *If D supports the operations specified in Section 7.2, Algorithm 1 returns α and makes $U - n + \alpha \leq U$ calls to* Increase*.*

Proof. We prove by induction that after each processed position t, every open interval i has $k_i = F_i(t)$, with $f_i[0] = \ell_i$ and each $f_i[q]$, $1 \leq q \leq k_i$, equal to the first position where F_i reaches q. The invariant holds when $\star$ is initialised and when an input interval opens, since its subproblem is empty. An opening changes no other subproblem value.

When j closes, the invariant gives $c_j = 1 + F_j(r_j - 1)$. The query test derived above and (6) identify exactly the remaining open intervals whose values increase. Lemma 16 shows that each increase is by one. Appending r_j records the first position where the new value is reached. All earlier entries and all other arrays remain correct, preserving the invariant.

The final call Close($\star$) returns $F_\star(2n-1) = \alpha$. There is one Increase call per unit increase, giving $U - n + \alpha \leq U$ calls by Lemma 16. □

7.4 A data-structure trade-off

It remains to find the intervals in (7) without examining every open interval. The next lemma lets a query search intervals with equal k_i in increasing order of ℓ_i, stopping at the first failed test.

Lemma 18 (Monotone attainment positions)**.** *For the subproblem values in Algorithm 1, if i, h are open intervals, $\ell_i < \ell_h$, and both have attained $q \geq 0$, then $f_i[q] \leq f_h[q]$.*

Proof. For $q = 0$ this is the order of the starting positions. For $q > 0$, when h first attained q, every family available to h was also available to i. Both intervals were open then, so i attained q no later. □

To implement D, we combine searches within groups of intervals with equal k_i with additional trees for selected values of c_j. Each additional tree replaces the group searches for that value by a single search, but must be maintained as records change. We choose which trees to create to balance these costs.

Two search procedures. Maintain a balanced search tree G_k of the open intervals with $k_i = k$, ordered by ℓ_i, and an ordered dictionary of the nonempty groups. Within a group with $k \geq c_j - 1$, the index $q = (k - c_j) + 1$ is fixed. Lemma 18 makes the tested entries nondecreasing, so the qualifying intervals form a prefix of the group. Searching these prefixes examines the returned intervals and at most one failed test per group.

For selected integers $s \geq 1$, also maintain a balanced search tree H_s containing every open interval i with $s \leq k_i + 1$, ordered by $f_i[(k_i - s) + 1]$, with ties broken by a fixed order of the intervals. When H_{c_j} exists, a query instead searches its qualifying prefix, with at most one failed test in total. Each opening, closing, or increase changes only the affected interval's group membership and at most one entry in each maintained H_s. Delete the old entries before changing the record, then insert the new entries.

Choosing which trees to maintain. Let g be the current number of nonempty groups. At a query with $c_j = s$, create H_s, if absent, exactly when $s^2 \leq g$. Build it by scanning all open intervals and inserting the eligible entries. Once created, retain it for the rest of the sweep. Thus the procedure needs no advance bound on the number of groups.

Lemma 19 (Data-structure bound)**.** *Let $U = \sum_j c_j$ and let ρ be the maximum number of distinct current values k_i at a query. The operations specified in Section 7.2 can be implemented for Algorithm 1 in $\mathcal{O}(U\sqrt{\rho}\log n)$ total time, using $\mathcal{O}(\alpha d)$ words for D and the query result list.*

Proof. The two searches return exactly the intervals in (7), without changing their records or returning any interval twice. Together with the record updates, this gives the operations required by the induction in Lemma 17.

Time. A tree H_s is created only when $s^2 \leq g \leq \rho$, so at most $\lfloor\sqrt{\rho}\rfloor$ trees are ever created. Each is built once in $\mathcal{O}(n \log n)$ time. The $\mathcal{O}(n + U)$ record updates each cost $\mathcal{O}(\sqrt{\rho}\log n)$ amortised time, including array growth. Since $n \leq U$, construction and updates together take $\mathcal{O}(U\sqrt{\rho}\log n)$ time.

Let z_j count the intervals returned by a query. Searching H_{c_j} costs $\mathcal{O}((z_j + 1)\log n)$ time. Otherwise, the group searches cost $\mathcal{O}((g + z_j)\log n)$. The creation rule gives $g < c_j^2$, and hence $g = \sqrt{g} \cdot \sqrt{g} < c_j\sqrt{\rho}$. Either search therefore costs $\mathcal{O}((c_j\sqrt{\rho} + z_j)\log n)$. Now $\sum_j c_j = U$ and $\sum_j z_j \leq U$ by Lemma 17, giving $\mathcal{O}(U\sqrt{\rho}\log n)$ total query time.

Space. Each open interval stores $k_i + 1$ array entries and has at most $k_i + 1$ entries across the trees H_s. There are at most $d + 1$ open intervals, each with $k_i \leq \alpha$. The groups, tree headers, lookup pointers, and query result list require $\mathcal{O}(n)$ further words. Thus the total is $\mathcal{O}(n + \alpha d) = \mathcal{O}(\alpha d)$, since $n \leq \alpha d$. □

Appendix A gives the detailed pseudocode and verifies its implementation of these operations.

Proof of Theorem 3. Lemma 17 proves correctness of Algorithm 1 when D supplies the specified operations, and Lemma 19 provides such an implementation.

Time. Let ρ be the maximum number of distinct current values k_i at a query. There are at most $d + 1$ open intervals, with values in $\{0, \ldots, \alpha\}$, so $\rho \leq \min\{d + 1, \alpha + 1\}$. Lemma 19 gives $\mathcal{O}(U\sqrt{\rho}\log n)$ time for the data-structure operations. The remaining work takes $\mathcal{O}(n + U)$ time by Lemma 17. With $n \leq U \leq \alpha d$ from Lemma 16, this gives

$$\mathcal{O}\Big(\alpha d\sqrt{\min\{\alpha, d\}}\log n\Big) = \mathcal{O}\Big(\min\{\alpha^{3/2}d, \alpha d^{3/2}\}\log n\Big).$$

Space. Lemma 19 gives $\mathcal{O}(\alpha d)$ words for the data structure. The remaining $\mathcal{O}(n)$ words are absorbed since $n \leq \alpha d$. The reconstruction described in Appendix A returns a maximum independent set within the same time and space bounds.

All counts, end-points, and record addresses fit in $\mathcal{O}(\log n)$ bits because $U \leq n^2$. The operations are comparisons, integer indexing, and balanced-tree operations. □

Alternating steps of Algorithm 1 and Nash and Gregg's algorithm [15] until either finishes gives $\widetilde{\mathcal{O}}(n + \mathcal{T}(n, \alpha, d))$ time without knowing the parameters in advance. Their algorithm uses $\mathcal{O}(n)$ space, so the combined space remains $\mathcal{O}(\alpha d)$.

8 Future work and open questions

Theorem 4 leaves some feasible parameter scales unresolved. A natural next case is $\alpha = d = \Theta(n^{3/4})$, where the known bound is $\mathcal{O}(n^{7/4})$. Extending the clique-to-cycle construction [14] to longer tuples may give a matching lower bound under the Combinatorial Clique Hypothesis.

Our parameter-sensitive lower bounds apply only to combinatorial algorithms. Lower bounds for arbitrary word-RAM algorithms, for example under the Strong Exponential Time Hypothesis (SETH) or an Orthogonal Vectors hypothesis, would strengthen the results. Our reductions also give only additive objective gaps, so they do not establish constant-factor approximation hardness. It remains of interest whether circle-graph MIS admits a truly subquadratic constant-factor approximation.

Another direction is to determine the dependence on the number e of edges in the represented graph. Circle-graph recognition takes $\mathcal{O}(n + e)$ time [17]. Understanding the complexity of MIS in terms of n and e, even when a chord model is supplied, would complement the bounds in terms of α and d.

Note

Yes, dear reader, this paper was written before ETH fell.

References

[1] Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. If the current clique algorithms are optimal, so is Valiant's parser. *SIAM Journal on Computing*, 47(6):2527–2555, 2018. Preliminary version in FOCS 2015. doi:10.1137/16M1061771.

[2] Hüseyin Acan. On a uniformly random chord diagram and its intersection graph. *Discrete Mathematics*, 340(8):1967–1985, 2017. doi:10.1016/j.disc.2016.11.004.

[3] Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *Journal of the ACM*, 42(4):844–856, 1995. doi:10.1145/210332.210337.

[4] Alberto Apostolico, Mikhail J. Atallah, and Susanne E. Hambrusch. New clique and independent set algorithms for circle graphs. *Discrete Applied Mathematics*, 36(1):1–24, 1992. doi:10.1016/0166-218X(92)90200-T.

[5] Alberto Apostolico, Mikhail J. Atallah, and Susanne E. Hambrusch. Erratum: New clique and independent set algorithms for circle graphs. *Discrete Applied Mathematics*, 41(2):179–180, 1993.

[6] Takao Asano, Hiroshi Imai, and Akira Mukaiyama. Finding a maximum weight independent set of a circle graph. *IEICE Transactions on Fundamentals*, E74-A(4):681–683, 1991.

[7] Prosenjit Bose, Paz Carmi, J. Mark Keil, Anil Maheshwari, Saeed Mehrabi, Debajyoti Mondal, and Michiel Smid. Computing maximum independent set on outerstring graphs and their relatives. *Computational Geometry*, 103:101852, 2022. doi:10.1016/j.comgeo.2021.101852.

[8] Mark Alan Buckingham. *Circle Graphs*. PhD thesis, New York University, 1981. URL: https://cs.nyu.edu/dynamic/reports/?year=1981.

[9] Yi-Jun Chang. Hardness of RNA folding problem with four symbols. In *27th Annual Symposium on Combinatorial Pattern Matching (CPM 2016)*, volume 54 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 13:1–13:12, 2016. doi:10.4230/LIPIcs.CPM.2016.13.

[10] Jimmy Ka Ho Chiu and Yi-Ping Phoebe Chen. Efficient conversion of RNA pseudoknots to knot-free structures using a graphical model. *IEEE Transactions on Biomedical Engineering*, 62(5):1265–1271, 2015. doi:10.1109/TBME.2014.2375360.

[11] Mina Dalirrooyfard and Virginia Vassilevska Williams. Conditionally optimal approximation algorithms for the girth of a directed graph. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, volume 168 of *LIPIcs*, pages 35:1–35:20, 2020. doi:10.4230/LIPIcs.ICALP.2020.35.

[12] Christine Eisenbeis, Sylvain Lelait, and Bruno Marmol. The meeting graph: A new framework for loop register allocation. Technical Report RR-2758, INRIA, 1995. URL: https://inria.hal.science/inria-00073934.

[13] Fănică Gavril. Algorithms for a maximum clique and a maximum independent set of a circle graph. *Networks*, 3(3):261–273, 1973. doi:10.1002/net.3230030305.

[14] Andrea Lincoln, Virginia Vassilevska Williams, and R. Ryan Williams. Tight hardness for shortest cycles and paths in sparse graphs. In *Proceedings of the Twenty-Ninth Annual ACM–SIAM Symposium on Discrete Algorithms (SODA 2018)*, pages 1236–1252. SIAM, 2018. doi:10.1137/1.9781611975031.80.

[15] Nicholas Nash and David Gregg. An output sensitive algorithm for computing a maximum independent set of a circle graph. *Information Processing Letters*, 110(16):630–634, 2010. doi:10.1016/j.ipl.2010.05.016.

[16] Ruth Nussinov, George Pieczenik, Jerrold R. Griggs, and Daniel J. Kleitman. Algorithms for loop matchings. *SIAM Journal on Applied Mathematics*, 35(1):68–82, 1978. doi:10.1137/0135006.

[17] Christophe Paul and Ignaz Rutter. Circle graphs can be recognized in linear time. In *43rd International Symposium on Theoretical Aspects of Computer Science (STACS 2026)*, volume 364 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 72:1–72:20, 2026. doi:10.4230/LIPIcs.STACS.2026.72.

[18] Kenneth J. Supowit. Finding a maximum planar subset of a set of nets in a channel. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 6(1):93–94, 1987. doi:10.1109/TCAD.1987.1270250.

[19] Gabriel Valiente. A new simple algorithm for the maximum-weight independent set problem on circle graphs. In *Algorithms and Computation, 14th International Symposium, ISAAC 2003*, volume 2906 of *Lecture Notes in Computer Science*. Springer, 2003. doi:10.1007/978-3-540-24587-2_15.

A Implementation of the joint-parameter algorithm

We give pseudocode for the data structure of Section 7.4. Algorithm 2 maintains the records and search trees. Algorithm 3 implements the two prefix searches and the rule for creating H_s. Created trees H_s are retained in a list in increasing order of s for updates, and located by an array of n pointers indexed by s.

The internal procedures AddEntries and DeleteEntries update the search trees without changing the stored record. Initially D and $\mathcal{G}$ are empty, $g = 0$, and no trees H_s exist.

Algorithm 2 Maintaining the representations f_i for open intervals.

Shared data for Algorithms 2 and 3:
Each interval's record stores f_i. Throughout, $k_i = |f_i| - 1$.
G_k, $\mathcal{G}$, and H_s are balanced binary search trees.
G_k: map $\ell_i \mapsto i$ for open intervals i with $k_i = k$, ordered by ℓ_i.
$\mathcal{G}$: map $k \mapsto G_k$ for nonempty groups, ordered by k, with $g = |\mathcal{G}|$.
H_s (if created): set of keys $(f_i[(k_i - s) + 1], i)$ for open intervals i with $1 \le s \le k_i + 1$.
H_s is ordered by attainment position, then by a fixed order of the intervals.

Input: The starting positions ℓ_i, including $\ell_\star = -1$.

procedure Open(i)
  Create a record for i with $f_i \leftarrow (\ell_i)$ in D
  AddEntries(i)

procedure Increase(i, t)
  DeleteEntries(i)
  Append t to f_i
  AddEntries(i)

procedure Close(i)
  $k \leftarrow k_i$, DeleteEntries(i)
  Discard the record of i, including f_i
  return k

procedure AddEntries(i)
  if G_{k_i} does not exist **then**
    Create empty G_{k_i}, insert $k_i \mapsto G_{k_i}$ into $\mathcal{G}$, $g \leftarrow g + 1$
  Insert entry $\ell_i \mapsto i$ into G_{k_i}
  for each s with $1 \le s \le k_i + 1$ and H_s created, in increasing order **do**
    Insert key $(f_i[(k_i - s) + 1], i)$ into H_s

procedure DeleteEntries(i)
  Delete the entry with key ℓ_i from G_{k_i}
  if G_{k_i} is empty **then**
    Delete key k_i from $\mathcal{G}$ and discard G_{k_i}, $g \leftarrow g - 1$
  for each s with $1 \le s \le k_i + 1$ and H_s created, in increasing order **do**
    Delete key $(f_i[(k_i - s) + 1], i)$ from H_s

Figure 6 illustrates the prefix searched within one group.

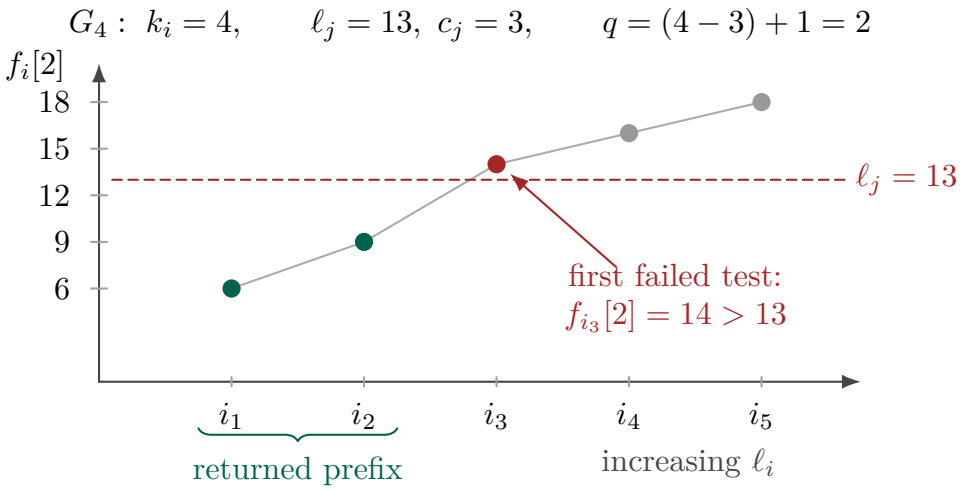


Figure 6: The query QUERY(j) within one group, with attainment positions $6, 9, 14, 16, 18$ in increasing order of the starting positions ℓ_i. The test is $f_i[2] < 13$. The first two intervals qualify. The third fails, and monotonicity excludes all later intervals without visiting them. Searching H_{c_j} uses the same prefix rule after ordering eligible intervals from all groups by $f_i[(k_i - c_j) + 1]$.

Algorithm 3 Finding the open intervals whose values increase when j closes.

Input: The shared trees of Algorithm 2, ℓ_j, and the computed weight $c_j \geq 1$.

```
1:  procedure QUERY(j)
2:    if H_{c_j} does not exist and c_j^2 ≤ g then
3:      Create empty H_{c_j}
4:      for each open interval i in D do
5:        if 1 ≤ c_j ≤ k_i + 1 then
6:          Insert key (f_i[(k_i − c_j) + 1], i) into H_{c_j}
7:      Insert H_{c_j} into the list in weight order, and store its pointer at index c_j
8:    L ← an empty list
9:    if H_{c_j} exists then
10:     for each key (a, i) in H_{c_j}, in increasing key order do
11:       if a ≥ ℓ_j then break
12:       Append i to L
13:   else
14:     for each key k ∈ G with c_j − 1 ≤ k, in increasing order do
15:       q ← (k − c_j) + 1
16:       for each interval i in G_k, in increasing order of ℓ_i do
17:         if f_i[q] ≥ ℓ_j then break
18:         Append i to L
19:   return L
```

Verification of the implementation. Correctness. We induct over the sweep, assuming the preceding operations satisfy the specification in Section 7.2. By Lemma 17, the arrays then contain the correct attainment positions. We verify the record updates and the query test (7). The internal procedures maintain every group G_k, the counter g, and the entries of every created set H_s. In particular, INCREASE deletes the old search entries before changing the record, then adds the entries for its new value and attainment position. OPEN initialises the record, and CLOSE returns its value before discarding it. The construction loop in QUERY(j) inserts exactly the required entries into a new H_{c_j}. Each key (a, i) in H_{c_j} satisfies $1 \leq c_j \leq k_i + 1$ and $a = f_i[(k_i - c_j) + 1]$. Thus the loop at line 10 returns exactly the qualifying prefix of H_{c_j}.

Without H_{c_j}, only groups with $k \geq c_j - 1$ can qualify, giving $0 \leq q = (k - c_j) + 1 \leq k$. Every

interval in G_k has the required entry $f_i[q]$, including $f_i[0] = \ell_i$ when $q = 0$. Lemma 18 makes these positions nondecreasing in the group's order. Consequently, the qualifying intervals form a prefix, and the loop at line 16 stops correctly at the first failed test. Every interval belongs to exactly one group and has at most one entry in H_{c_j}, so no interval is returned twice. The records do not change during a query. Thus it satisfies the specification, completing the induction.

Implementation costs. The ordered dictionary $\mathcal{G}$ permits searching only nonempty groups with $k \geq c_j - 1$. The list of created trees and their lookup array allow record updates and queries to find the required H_s within the costs used in Lemma 19. Each tree operation costs $\mathcal{O}(\log n)$, and doubling array capacity when full makes appending amortised constant time. The two search loops inspect only their returned prefixes and the first failed test, when present.[3] The time and space analysis in Section 7.4 therefore applies to this implementation. □

Recovering a maximum independent set. Alongside each entry $f_i[q]$, store a pointer to a representation of an independent family of size q, using an empty representation for $q = 0$. When j closes, retain a representation S_j consisting of j and the family represented by its last array entry. Its size is c_j. When a query for j increases k_i from k to $k + 1$, put $q = k - c_j + 1$. The recurrence and the unit-increase property give $F_i(\ell_j - 1) = q$. Store, for the new entry, a pair of pointers to the family at entry q and to S_j. The first family ends before ℓ_j, and S_j consists of j and intervals inside it. Their union is therefore independent and has size $q + c_j = k + 1$.

These representations are immutable: closing an interval discards its search record but not any representation referred to elsewhere. There is one new representation per input interval and per increase, so they occupy $\mathcal{O}(n + U) = \mathcal{O}(\alpha d)$ words and add constant work per operation. Following the final pointer for $\star$ reports the maximum independent family. Each combination joins disjoint families, so no interval is reported twice and the traversal takes $\mathcal{O}(\alpha)$ time. The pseudocode above omits these pointers.

[3]Minimum pointers and links between consecutive entries remove the logarithmic factors from query searches.