New Algorithms For Maximum Independent Sets of Circle Graphs

Nicholas Nash* David Gregg†

April 8, 2013

Abstract

We describe a new algorithm for computing a maximum independent set of an unweighted circle graph, where the chords or intervals representing the graph may share end-points. The algorithm requires $O(m + n \min\{\alpha^2, m\})$ time and $O(m)$ space, where $\alpha \leq n$ is the independence number of a circle graph represented by m intervals with n end-points. The best previously described algorithm required $O(nm)$ time and $O(n^2)$ space. We also describe an improved algorithm for the weighted case.

1 Introduction

A circle graph is the intersection graph of a finite set of chords in a circle. Circle graphs find applications in a number of areas, including in register allocation algorithms in optimizing compilers [6, 12], bioinformatics [14] and VLSI applications [13, 5] where the maximum independent set problem arises.

A series of polynomial time algorithms for computing a maximum independent set of a circle graph have been described [7, 13, 3, 9, 1, 2, 17, 11]. The best known algorithms [17, 11] require $O(n)$ space and operate in $O(nd)$ time and $O(n \min\{d, \alpha\})$ time in the weighted and unweighted cases respectively. Here α is the independence number of the circle graph and $d = O(n)$ is a parameter known as the density.

Some other NP-Complete problems on general graphs have efficient algorithms for circle graphs, notably maximum clique [15], whereas problems such as computing the chromatic number or minimum dominating set are hard for both circle graphs as well as for general graphs [16, 10]. The best known algorithm for recognizing circle graphs requires $O(n+m)$ time [8] (excluding a factor of the inverse Ackermann function).

*Susquehanna International Group, nicholas.nash@gmail.com
†Lero, Trinity College Dublin, dgregg@cs.tcd.ie.

Almost all algorithms for computing a maximum independent of a circle graph have operated on a representation of the graph as a set of n intervals on the real line. Figure 1 illustrates the equivalence of the representation of a circle graph as chords in a circle or as intervals on the real line. Without loss of generality, it can be assumed that no two intervals share an end-point, because an equivalent representation of the circle graph may easily be constructed by slightly moving the end-points of the intervals, as is illustrated by Gavril [7].

Recently Bonsma and Breuer [4] noted a simple $O(nm)$ time algorithm for the case where n interval end-points may be shared by up to $m = O(n^2)$ intervals. In addition, they noted that in the case of a dense interval model of $m = \Omega(n^2)$ intervals, transforming to an interval representation with distinct end-points and using an existing maximum independent set algorithm can be less efficient than their algorithm, requiring $\Theta(n^4)$ time.

In this paper, we describe a simple algorithm that improves on this time complexity in the case of unit edge weights, requiring $O(m + n\min\{\alpha^2, m\})$ time and $O(m)$ space. Note that $\alpha \leq n$ and that this new algorithm is up to a factor of n times more efficient than the best previous algorithm for dense interval representations of $m = \Omega(n^2)$ intervals where $\alpha = O(\sqrt{n})$. We begin by describing an existing algorithm in Section 2. We then describe an improved algorithm for weighted circle graphs. Finally we present an improved algorithm for the unweighted case in Section 4 that is the main result of this paper.

2 An Existing Algorithm for Weighted Circle Graphs

We refer to a set I of m intervals with end-points in $\{1, \ldots, n\}$ as an $I(n, m)$-representation of a circle graph, and assume that $m \geq n$. We focus on describing algorithms for computing the cardinality of a circle graph's maximum independent sets. The maximum independent sets themselves can be maintained without worsening the time complexities of the algorithms, in a manner analogous to the distinct end-point case [17, 11]. For a given $I(n, m)$-representation, we denote the set of intervals with left end-point q and right end-point at most r by $L_{q,r}(I)$. For a set of intervals S, we denote the cardinality of their maximum independent sets by $MIS[S]$. We denote by $I_{q,r}$ the set of all intervals in I contained in $[q, r]$. Given an interval $i = [a, b]$ we write the cardinality of the contained maximum independents sets as $CMIS[i] = MIS[I_{a+1,b-1}]$. Throughout this paper, we make use of the following quantity:

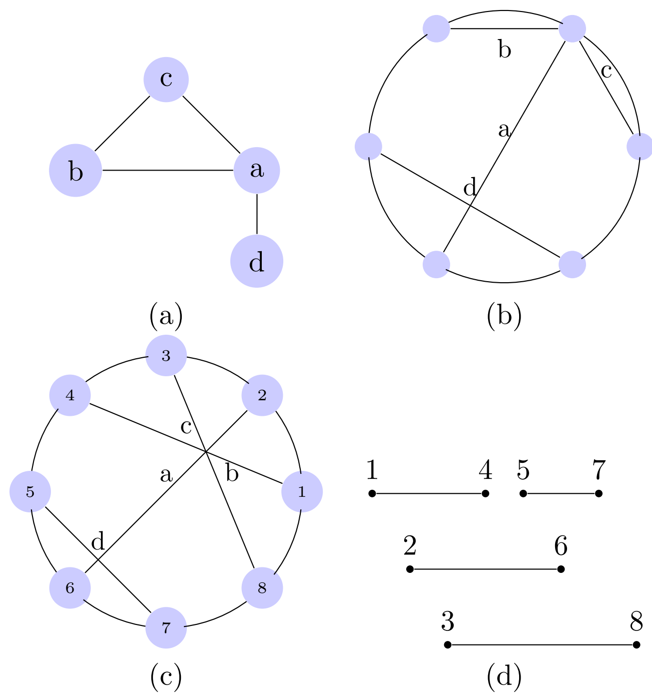

Figure 1: A circle graph (a) together with its chord model in (b). The equivalent distinct end-point chord model is shown in (c), with its associated interval representation in (d)

Definition 2.1 *Given the $I(n,m)$-representation of a circle graph and a set of intervals S*

$$U_r(S,I) = \max_{i=[a,b]\in S}\{1 + CMIS[i] + MIS[I_{b+1,r}]\}$$

We make use of the following recurrence for computing maximum independent sets given $I(n,m)$-representation of a circle graph:

$$MIS[I_{q,r}] = \max\{MIS[I_{q+1,r}], U_r(L_{q,r}(I), I)\} \tag{1}$$

Previous algorithms for computing the cardinality of maximum independent sets of circle graphs [17, 11] have made use of recurrences analagous to Recurrence 1. However, with the exception of Bonsma and Breuer's [4] algorithm, previous algorithms have dealt with the case where intervals do not share end-points, implying that $U_r(S,I)$ can be evaluated in constant time for any value of r.

Evaluating Recurrence 1 directly requires $O(nm)$ time: r is increased from 1 up to n, and for each value of r $MIS[I_{q,r}]$ is evaluated from $q = r$ down to 1. Hence, for a given value of r $O(\sum_{q=1}^{r} |L_{q,r}(I)|) = O(m)$ time is spent. We note that this is the algorithm described by Bonsma and Breuer [4], although their order

Input: An $I(n,m)$-representation of a circle graph.
Output: The cardinality of the maximum independent sets contained in each interval.

```
M[1...n] ← 0
C[1...n] ← 0
for r ← 1 to n do
    if there are intervals with right end-point r then
        let i = [a, r] be the longest such interval
        for q ← r − 1 downto a do
            M[q] ← M[q + 1]
            foreach interval j = [q, b] with b ≤ r do
                if 1 + CMIS[j] + M[b + 1] > M[q] then
                    M[q] ← 1 + C[j] + M[r + 1]}
            if q is the left end-point of an interval i = [q, r] then
                C[i] ← M[q + 1]
```

Figure 2: An $O(\ell_s + C)$ time and $O(m)$ space algorithm for computing the $CMIS$ values of a circle graph.

of evaluation is different.

3 An Improved Algorithm for Weighted Circle Graphs

As Valiente [17] noted in the distinct end-point case, evaluating $MIS[I_{q,r}]$ for every pair of q and r is unnecessary, since only those subproblems defined by interval end-points are re-used. That is, the problem can be reduced to computing the value of $CMIS$ for each interval. Moreover we note that when the value of $CMIS$ is known for each interval, the weight of the circle graph's maximum independent sets can be determined using only a further $O(n)$ time [17].

Computing the value of MIS only at sub-problems defined by interval end-points reduces the space required from $O(n^2)$ to $O(m)$. In addition, we note that it also reduces the time required to evaluate the recurrence. This is a result of the fact that if there are two intervals $i = [a,b], j = [c,b]$, with $a < c$ then $CMIS[j] = MIS[I_{c+1,b-1}]$ will be available when computing $CMIS[i]$. Thus, when the simplified version of Valiente's algorithm found in [12] is adapted to the shared end-point case, the evaluation of $CMIS$ is only required for the longest interval with a given right end-point. This is because the other intervals sharing their right end-point with this longest interval can have their $CMIS$ values filled in during the evaluation of the longest interval's $CMIS$ value. Figure 2 illustrates the resulting algorithm for computing the $CMIS$ values of a shared end-point interval representation.

In the distinct end-point case Valiente's algorithm requires $O(\ell)$ time, where $\ell \leq nd$ is the sum of the lengths of all the intervals in the interval representation of the circle graph, and d is the largest number of intervals crossing any point on the interval representation.

Let l_q be the length of the longest interval with right end-point q, and $\ell_s = \sum_{q=1}^{n} l_i$. Moreover define c_q as the number of intervals contained in the longest interval with right end-point q, and $C = \sum_{q=1}^{n} c_q$. In the shared end-point case, this adaptation of Valiente's algorithm requires time $O(\ell_s + C)$. For weighted circle graphs, this algorithm is the most efficient in both shared end-point and distinct end-point interval models.

In the distinct end-point case, Valiente notes that $\ell \leq nd$. In the shared end-point case, the analogous parameter, d_s, is the largest number of intervals with distinct right end-points crossing any point in the interval representation. Note $\ell_s \leq nd_s$, $d_s \leq m$ and clearly $C \leq nm$. Thus the algorithm's $O(nd_s + C)$ time is potentially superior to the direct $O(nm)$ time algorithm. We also note that when there are no shared end-points the definitions are such that $\ell_s = \ell$ and $d_s = d$; moreover $C = O(\ell)$ in the distinct end-point case.

4 Improved Algorithms for Unweighted Circle Graphs

In this section we describe improved algorithms for computing a maximum independent set of a circle graph, by noting that the number of distinct values of $CMIS$ may be considerably less than the number of intervals m in the circle graph's interval representation. We now note the key insights that lead to the improved algorithms.

Definition 4.1 *Given an $I(n,m)$-representation of a circle graph, the left exclusion set at q, $1 \leq q < r \leq n$ is*

$$X_{q,r}(I) = \{i = [q,s] \in I_{q,r} \mid \exists j = [q,s'] \text{ s.t. } CMIS[j] \geq CMIS[i] \text{ and } s' < s\}$$

Lemma 4.2 *Given an $I(n,m)$-representation of a circle graph with independence number α*

$$|L_{q,r}(I) \setminus X_{q,r}(I)| \leq \alpha$$

for $1 \leq q < r \leq n$

Proof If $i, j \in L_{q,r}$ and $CMIS[i] = CMIS[j]$ then either $i \in X_{q,r}$ or $j \in X_{q,r}$. Thus for any $k \in L_{q,r} \setminus X_{q,r}$ $\nexists p \in L_{q,r} \setminus X_{q,r}$ with $CMIS[k] = CMIS[p]$. Since there can only be at most α distinct values of $CMIS$ the result follows.

Lemma 4.3 *Given an $I(n,m)$-representation of a circle graph*

$$MIS[I_{q,r}] = MIS[I_{q,r} \setminus X_{q,r}(I)]$$

for $1 \leq q \leq r \leq n$

Proof If q is the right end-point of an interval then $X_{q,r} = \emptyset$ and the result follows. If q is the left end-point of an interval then

$$MIS[I_{q,r}] = \max\{MIS[I_{q+1,r}], U_r(L_{q,r}(I), I)\}$$

and noting that $X_{q,r} \cap I_{q+1,r} = \emptyset$

$$MIS[I_{q,r} \setminus X_{q,r}] = \max\{MIS[I_{q+1,r}], U_r(L_{q,r}(I) \setminus X_{q,r}(I), I)\}$$

In the case where $MIS[I_{q,r}] = MIS[I_{q+1,r}]$ then

$$MIS[I_{q+1,r}] \geq U_r(L_{q,r}(I), I) \geq U_r(L_{q,r} \setminus X_{q,r}(I), I)$$

and the result follows.

In the case where $MIS[I_{q,r}] = U_r(L_{q,r}(I), I) = 1 + CMIS[i] + MIS[I_{a+1,r}]$ for some $i = [q,a] \in L_{q,r}(I)$. If $i \notin X_{q,r}(I)$ then $U_r(L_{q,r}(I), I) = U_r(L_{q,r}(I) \setminus X_{q,r}(I), I)$ and the result follows.

In the final case, where $i \in X_{q,r}(I)$ then $\exists j = [q,b] \in L_{q,r}(I) \setminus X_{q,r}(I)$ such that $CMIS[j] \geq CMIS[i]$ and $b < a$. Since $b < a$, $MIS[I_{b+1,r}] \geq MIS[I_{a+1,r}]$. Thus $U_r(L_{q,r}(I) \setminus X_{q,r}(I), I) \geq U_r(L_{q,r}(I), I)$, and since $U_r(L_{q,r}(I), I) \geq U_r(L_{q,r}(I) \setminus X_{q,r}(I), I)$ the result follows.

The preceding Lemma leads directly to an improved algorithm for computing the cardinality of a maximum independent set of a circle graph.

Lemma 4.4 *Given the $I(n,m)$-representation of a circle graph with independence number α, the cardinality of its maximum independent sets can be computed in $O(n \min\{n\alpha, m\})$ time and $O(m)$ space.*

Proof Let $B_{q,r}(I)$ be the largest value of $CMIS[i]$ for $i \in L_{q,r}(I)$. We refer to the algorithm of Figure 3, and firstly show that it correctly evaluates the cardinality of the maximum independent sets of the circle graph.

Assume that at the beginning of iteration r of the outer-loop of the algorithm, for $1 \leq q < r$, $LX[q] = L_{q,r-1}(I) \setminus X_{q,r-1}(I)$, that $M[q] = MIS[q, r-1]$, and finally that $MLX[q] = B_{q,r-1}(I)$. It is then clear that, as the algorithm decreases q from $r-1$ to 1 it sets $M[q]$ to $\max\{MIS[I_{q+1,r} \setminus X_{q,r}(I)], U_r(L_{q,r}(I) \setminus X_{q,r}(I), I)\}$, that is, to $MIS[I_{q,r} \setminus X_{q,r}(I)] = MIS[I_{q,r}]$, by Lemma 4.3.

After this, the algorithm updates $MLX[q]$, only if there is an interval $i \in L_{q,r}$ with $CMIS[i] > MLX[q]$, and in this case it adds the interval to $LX[q]$. Since as the algorithm iterates r from 1 to n, it encounters intervals in $I_{1,n}$ in increasing order of right end-point, the algorithm adds i to $LX[q]$ only if there is no interval $j = [q, r']$ with $r' < r$ and $CMIS[j] \geq CMIS[i]$, that is, it adds i to $LX[q]$ only if $i \notin X_{q,r}(I)$.

Thus, at the end of the iteration of the loop of q from r down to 1, it is clear that the assumptions made at the beginning of this lemma now hold for the additional value of $q = r$. Since the assumptions are trivially fulfilled before the beginning of the algorithm, it correctly evaluates the cardinality of the circle graph's maximum independent sets.

For the time complexity, note that by Lemma 4.2 at most $O(\alpha)$ time is spent for any value of q, and thus for all values of q at most $O(n^2\alpha)$ time is spent. Clearly, the algorithm can also spend no more than $O(\sum_{q=1}^{r} |L_{q,r}|) = O(m)$ time for each value of r, and the claimed time complexity follows of $O(n \min\{n\alpha, m\})$ follows. The space required follows from that required for storing $CMIS$ values for each of the m intervals in C.

The time complexity of the algorithm of Figure 3 can be further improved by making use of the following observation: As r increases to $r+1$, the set of values in the array M that increase may be substantially smaller than the total number of values examined in the right-to-left iteration of q from $r-1$ down to 1. This is exactly the observation used in the case of distinct end-points that gives rise to an $O(n\alpha)$ time algorithm from an $O(n^2)$ time algorithm [11], from where we obtain the following definition, modified to the shared end-point case:

Definition 4.5 *Given the $I(n,m)$-representation of a circle graph, the update set at r, $1 \leq r \leq n$, is*

$$S_r = \{q \mid MIS[I_{q,r}] > MIS[I_{q,r-1}] \text{ for } 1 \leq q \leq r\}$$

We also define the equivalent of $X_{q,r}$ above as

Input: An $I(n,m)$-representation of a circle graph.
Output: The cardinality of the circle graph's maximum independent sets.

```
M[1...n] ← 0
C[1...m] ← 0
MLX[1...n] ← 0
LX[1...n] ← ∅
for r ← 1 to n do
    if there are intervals with right end-point r then
        for q ← r − 1 downto 1 do
            M[q] ← M[q + 1]
            foreach interval j = [q, b] in LX[q] do
                if 1 + CMIS[j] + M[b + 1] > M[q] then
                    M[q] ← 1 + C[j] + M[r + 1]}
            if q is the left end-point of an interval i = [q, r] then
                C[i] ← M[q + 1]
                if C[i] > MLX[q] then
                    C[i] ← MLX[q]
                    LX[q] ← LX[q] ∪ {i}
```

Figure 3: An improved algorithm for computing the cardinality of a maximum independent set of a circle graph, requiring $O(n \min\{n\alpha, m\})$ time and $O(m)$ space.

Definition 4.6 *Given an $I(n,m)$-representation of a circle graph, the right exclusion set at q, $1 \leq q < r \leq n$ is*

$$X'_{q,r}(I) = \{i = [s,q] \in I \mid \exists j = [s',q] \text{ s.t. } CMIS[j] \geq CMIS[i] \text{ and } s < s'\}$$

With an argument precisely analagous to the one given in Lemma 4.3, it follows that $MIS[I_{q,r}] = MIS[I_{q,r} \setminus X'_{q,r}]$, The algorithm of Figure 4 computes $MIS[I_{q,r} \setminus X'_{q,r}]$, but, for each value of r, it updates a cell $M[q]$ and pushes it onto the stack T only if $q \in S_r$. Note, that in the PREPARE LOOP of the algorithm, only the indices of cells whose value has increased as a result of new intervals contained in $I_{q,r}$ but not $I_{q,r-1}$ are pushed onto the stack T. The STACK LOOP of the algorithm then, making use of Recurrence 1 determines the set of cells whose value may increase due to an increase in value of the cells on the stack T. We omit the proof of correctness of the algorithm in Figure 4, as it is exactly analagous to the one for the $O(n\alpha)$ time algorithm in the distinct end-point case [11].

Lemma 4.7 *Given the $I(n,m)$-representation of a circle graph, the algorithm of Figure 4 operates in $O(m + n\alpha^2)$ time and $O(m)$ space.*

Proof (Sketch) The total time spent by the PREPARE LOOP of the algorithm across all values of r is $O(m)$, moreover for each value of q, this loop ensures $RX[q]$ contains $I_{q,r} \setminus X'_{q,r}$. For each value of r, the stack T contains only indices $x \in S_r$. For any iteration of the STACK LOOP, at most $O(\alpha)$ time is spent pushing new

Input: An $I(n, m)$-representation of a circle graph.
Output: The cardinality of the circle graph's maximum independent sets.

```
M[1...n] ← 0
C[1...m] ← 0
MRX[1...n] ← 0
RX[1...n] ← ∅
T ← empty stack
for r ← 1 to n do
    foreach interval i = [a, r] in increasing order of length (PREPARE LOOP)
        C[i] ← M[a + 1]
        if a > 1 and 1 + C[i] > M[a − 1] then
            M[a − 1] ← 1 + C[i]
            PUSH(T, a − 1)
        if C[i] > MRX[i] then
            MRX[i] ← C[i]
            RX[i] ← {i} ∪ RX[i]
    while T is not empty do (STACK LOOP)
        x ← POP(T)
        if x > 1 AND M[x] > M[x − 1] then
            PUSH(T, x − 1)
        foreach interval j = [p, x − 1] in RX[x − 1]
            if 1 + C[j] + M[x] > M[p]
                M[p] ← 1 + C[j] + M[x]
                PUSH(T, p)
```

Figure 4: An $O(m+n\min\{\alpha^2, m\})$ time and $O(m)$ space algorithm for computing the cardinality of a circle graph's maximum independent sets.

values onto the stack T, since, analagously to Lemma 4.2 $\left|I_{q,r} \setminus X'_{q,r}\right| \leq \alpha$. Note that the STACK LOOP can iterate in total at most $n\alpha$ times, as any more would increase a cell of M to a value greater than α. Hence the total time spent by the algorithm is $O(m+n\alpha^2)$. The extra space required by the algorithm is only that for C, MRX, RX and T which is clearly $O(m)$.

We finish by noting that this $O(m + n\alpha^2)$ time algorithm can terminate at any point when a cell of M increases beyond $\sqrt{m}$, and switch to one of the $O(nm)$ time algorithms described in Section 2. This gives an algorithm operating in $O(m + n\min\{\alpha^2, m\})$ time and $O(m)$ space.

5 Conclusion

This paper has described how to adapt the improved version of Valiente's [17] algorithm described in [12] to the case of shared interval end-points. We note that this is the most efficient algorithm for computing a maximum independent set of a weighted circle graph. In addition, we have described a new $O(m + n\min\{\alpha^2, m\})$ time and $O(m)$ space algorithm for computing a maximum independent set of an unweighted circle graph. The algorithm is simple, and does not require complicated data structures in order to run within the stated bounds. As a result of this as well as the improved time complexity, we believe the algorithm has a strong potential to offer improved performance in applications. We note that the algorithm is especially efficient in the case of dense interval representations.

Acknowledgements

We thank Oliver Nash for many useful discussions.

References

[1] A. Apostolico, M. J. Atallah, and S. E. Hambrusch. New clique and independent set algorithms for circle graphs. *Discrete Applied Mathematics*, 36(1):1–24, 1992.

[2] A. Apostolico, M. J. Atallah, and S. E. Hambrusch. Erratum: New clique and independent set algorithms for circle graphs (discrete applied mathematics 36 (1992) 1-24). *Discrete Applied Mathematics*, 41(2):179–180, 1993.

[3] T. Asano, H. Imai, and A. Mukaiyama. Finding a maximum weight independent set of a circle graph. *IEICE Transactions*, E74(4):681–683, 1991.

[4] P. Bonsma and F. Breuer. Counting hexagonal patches and independent sets in circle graphs. *Algorithmica*, 63(3):645–671, 2012.

[5] J. Cong and C. L. Liu. Over-the-cell channel routing. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 9(4):408–418, 1990.

[6] D. de Werra, C. Eisenbeis, S. Lelait, and B. Marmol. On a graph-theoretical model for cyclic register allocation. *Discrete Applied Mathematics*, 93(2-3):191–203, 1999.

[7] F. Gavril. Algorithms for a maximum clique and a maximum independent set of a circle graph. *Networks*, 3(3):261–273, 1973.

[8] E. Gioan, C. Paul, M. Tedder, and D. G. Corneil. Circle graph recognition in time $o(n+m)\alpha(n+m)$. *CoRR*, abs/1104.3284, 2011.

[9] O. Goldschmidt and A. Takvorian. An efficient algorithm for finding a maximum weight independent set of a circle graph. *IEICE Transactions*, E77-A(10):1672–1674, 1994.

[10] J. M. Keil. The complexity of domination problems in circle graphs. *Discrete Applied Mathematics*, 42(1):51–63, 1993.

[11] N. Nash and D. Gregg. An output sensitive algorithm for computing a maximum independent set of a circle graph. *Inf. Process. Lett.*, 110(16):630–634, 2010.

[12] N. Nash, S. Lelait, and D. Gregg. Efficiently implementing maximum independent set algorithms on circle graphs. *J. Exp. Algorithmics*, 13:1.9–1.34, 2009.

[13] K. J. Supowit. Finding a maximum planar subset of a set of nets in a channel. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 6(1):93–94, 1987.

[14] K. M. Swenson, Y. To, J. Tang, and B. M. E. Moret. Maximum independent sets of commuting and noninterfering inversions. *BMC Bioinformatics*, 10(S-1), 2009.

[15] A. Tiskin. Semi-local string comparison: algorithmic techniques and applications. *CoRR*, abs/0707.3619, 2013.

[16] W. Unger. On the k-colouring of circle-graphs. In *Proceedings of the 5th Annual Symposium on Theoretical Aspects of Computer Science*, STACS '88, pages 61–72, London, UK, UK, 1988. Springer-Verlag.

[17] G. Valiente. A new simple algorithm for the maximum-weight independent set problem on circle graphs. In T. Ibaraki, N. Katoh, and H. Ono, editors, *ISAAC*, volume 2906 of *Lecture Notes in Computer Science*, pages 129–137. Springer, 2003.