TR-88-014

# A SIMPLE ALGORITHM FOR THE MAXIMUM INDEPENDENT SET PROBLEM ON CIRCULAR-ARC GRAPHS

# A SIMPLE ALGORITHM FOR THE MAXIMUM INDEPENDENT SET PROBLEM ON CIRCULAR–ARC GRAPHS

Wen–Lian Hsu †

Department of Industrial Engineering
and Management Sciences
Northwestern University
Evanston, IL 60208

Key Words: graph, independent set, algorithm

## Abstract

An arc family F is a collection of arcs on the unit circle. An independent set of arcs in F is a set of pairwise nonoverlapping arcs in F. The maximum independent set (MIS) problem on F is to determine a maximum size independent set in F. Given that the endpoints of all arcs are sorted clockwise, Masuda and Nakajima gave an O(n) algorithm for the MIS problem based on the idea of reducing F to its proper subfamily. We simplify their approach by further reducing F to a strictly proper subfamily.

† This research was supported in part by the National Research Council of the Republic of China while the author visited Academia Sinica.

# A Simple Algorithm for the Maximum Independent Set Problem on Circular–Arc Graphs

## 1. Introduction

An arc family F is a collection of arcs on the unit circle. An independent set of arcs in F is a set of pairwise nonoverlapping arcs in F. The ***maximum independent set (MIS) problem*** on an arc family F is to determine a maximum size independent set in F. A graph G with vertex set V and edge set E is said to be a ***circular–arc graph*** if there exists an arc family F and a one–to–one correspondence between V and F such that two vertices in V are adjacent in G iff their corresponding arcs overlap.

Without loss of generality, assume all arc endpoints are distinct and no arc covers the entire circle. Now, label the n arcs arbitrarily from 1 through n. Label the endpoints from 1 to 2n according to their clockwise order. Denote an arc that begins at endpoint p and ends at endpoint q in the clockwise direction by (p,q). Define p to be the ***head*** (or counterclockwise endpoint) of the arc and q to be the ***tail*** (or clockwise endpoint). An example is shown in Figure 1. Define a ***head*** (respectively, ***tail***) ***block*** to be a set of maximally contiguous heads (respectively, tail) in the clockwise endpoint order. The term "block" is used to refer to either a head block or a tail block.

Figure 1. A circular–arc family F

The continuous part of the circle which begins with endpoints c and ends with d in the clockwise direction is referred to as ***segment*** (c,d) of the circle. We use "arc" to refer to a member of F and "segment" to refer to a part of the circle between two endpoints. By this definition, an arc (p,q) of F is also a segment (p,q). A point on the circle is said to be in arc (p,q) if it falls within the segment (p,q).

Two arcs that do not overlap with each other are said to be ***independent***. An arc i of F is said to be ***properly contained in*** another arc j if both endpoints of i are contained in arc j. An arc family F is said to be ***proper*** if no arc in F is properly contained in another. Given any arc family F, let C(F) be the collection of arcs which are properly contained in some other arc in F. Define $F' = F \backslash C(F)$ to be the proper subfamily of F. Clearly, F' is proper. Given any MIS P of F, one can find an independent set P' in F' of the same size by suitable exchange. Hence. $\alpha(F) = \alpha(F')$.

In [2], an O(n) time algorithm is given for the maximum independent set problem on circular–arc graphs with n vertices assuming that the endpoints of their corresponding arcs have already been sorted in the clockwise direction around the circle. The main steps in their algorithm are the following:

(1) Reduce the MIS problem on an arc family F to that of its proper subfamily F'.

(2) Determine a MIS in F' in O(n) time.

In this paper we further simplify their algorithm by removing more arcs in step 1 which are not crucial in forming an MIS of F so that step 2 becomes trivial. More precisely, define an arc i to be ***dominated*** by another arc j if every arc overlapping with j (including j itself) also overlaps with i. Clearly, if a MIS contains i, then one can replace i by j and obtain another MIS. Define an arc family F to be ***strictly proper*** if no arc in F is dominated by another. Clearly, if i contains j, then i is dominated by j. Hence, a strictly proper family is also proper. The main steps in our algorithm are the following:

(1) Reduce the MIS problem on an arc family F to that of a strictly proper subfamily F" such that $\alpha(F") = \alpha(F)$.

(2) Determine a MIS of F" in $O(|\alpha(F)|)$ time.

Before discussing the properties of strictly proper families, we describe some properties of proper families which tends to explain why the MIS problem is easy on them. For each arc i, define *PREV*(i) to be the arc whose tail is the first tail encountered in a counterclockwise traversal from h(i); define *SUCC*(i) to be the arc whose head is the first head encountered in a clockwise traversal from t(i); define *GD*(i) (GD is short for GREEDY) to be the maximal independent set of the form i, $i_1, \ldots, i_k$, where $i_1 = \mathrm{SUCC}(i)$ and $i_t = \mathrm{SUCC}(i_{t-1})$, $t = 2, \ldots, k$.

Lemma 2. *Let P be an independent set in a proper family F. Let i be any arc in P. Then* $|P| \le |GD(i)|$. *Furthermore, if j is in GD(i), then* $|GD(i)| \le |GD(j)| \le |GD(i)| + 1$.

Proof. Denote GD(i) by the set $\{i, i_1, \ldots, i_k\}$ as described above, where $k = |GD(i)|-1$. Denote P by the set $\{i, i_1', \ldots, i'_{|P|-1}\}$, in which the arcs are ordered according to their heads in the clockwise traversal from h(i). If every arc in P is also in GD(i), then we are done. Hence assume the contrary. Let $i_t'$ be the arc in $P \backslash GD(i)$ with the smallest index. Then because $i_t = \mathrm{SUCC}(i_{t-1})$, we must have $h(i_t) \in (t(i_{t-1}), h(i_t'))$. Because F is proper, $t(i_t)$ is in arc $i_t'$. Hence, $P_t = (P \cup \{i_t\}) \backslash \{i_t'\}$ is also an independent set with $|P_1| = |P|$. By induction, for each $s > t$, $P_s = (P \cup \{i_t, \ldots, i_s\}) \backslash \{i_t', \ldots, i_s'\}$ is an independent set of size $|P|$ and $H(i_s') \in (t(i_{s-1}), h(i_s'))$. Hence, $|P| - 1 \le |GD(i)| - 1$ and $|P| \le |GD(i)|$.

Now, assume j is in GD(i). Suppose $j = i_m$. We have $\{i_m, \ldots, i_k\} \subseteq GD(j)$. If $i \in GD(j)$, then one can easily verify that GD(i) = GD(j). Hence, assume $i \notin GD(j)$. Then, $GD(j) = \{i_m, \ldots, i_k, i_{k+1}, \ldots, i_{|GD(j)|+m-1}\}$, where $i_s = \mathrm{SUCC}(i_{s-1})$ for all $s \ge k+1$. Let $i_r$ be the arc of GD(i) with the smallest r that is common to both GD(i) and GD(j). Then $GD(i) \backslash GD(j) = \{i, i_1 \ldots, i_{r-1}\}$. Now, the arcs in $GD(j) \backslash GD(i)$ must have their heads contained in one of segment $(t(i_k), h(i))$ or arcs i, $i_1, \ldots, i_{r-1}$. Since F is proper, each of these segment or arcs can contain at most one head of GD(j).

Therefore, there can be at most r+1 such arcs and $|GD(j)\setminus GD(i)| \le r+1$. Hence, $|GD(j)| \le |GD(i)| + 1$. ∎

Lemma 3. *Let i be any arc in a proper family F. Let $P_{\max}$ be a MIS in F. Then $|GD(i)|$ is either $|P_{\max}|$ or $|P_{\max}| - 1$.*

Proof. If $GD(i) \cap P_{\max} = \emptyset$, then arcs in $P_{\max}$ must have their heads contained in one of the segment $(t(i_k),h(i))$ or arcs $a_1, i_1, \ldots, i_k$, where $k = |GD(i)|-1$. Hence, $|P_{\max}| \le |GD(i)| + 1$.

If $GD(i) \cap P_{\max} \ne \emptyset$, let j be a common element. By Lemma 2, $|P_{\max}| = |GD(j)| \le |GD(i)| + 1$. ∎

## 2. A characterization of strictly proper families

A strictly proper subfamily F" of F satisfying $\alpha(F") = \alpha(F)$ can be obtained iteratively as follows. Initially, let F be the current family. We then start removing dominated arcs in the current family one by one. This process terminates when there exists no dominated arc in the current family, which must then be strictly proper. It is clear that removing a dominated arc does not change the size of a MIS, since if a MIS contains an arc i which is dominated by j, then i can be replaced by j. Hence the final strictly proper subfamily F" obtained satisfies $\alpha(F")$ $= \alpha(F)$. Note that an arc which is not currently dominated by any other arcs may become dominated after some other arcs are removed. Since the order of arcs removed is far from unique, it is possible to obtain different strictly proper subfamilies at the end. An example is shown in Figure. 2.

Figure 2. Two different strictly proper subfamilies

A brute–force implementation of the above iterative algorithm could take

$O(n^2)$ time. In Section 3 we present an $O(n)$ algorithm for determining such an $F''$. The algorithm depends on the following characterization of a strictly proper family.

Theorem 2. *Let F be an arc family which is not a clique. Then F is strictly proper if and only if F is proper and each block of F* contains *exactly one endpoint.*

Proof. Assume F is strictly proper. Then F must be proper. Suppose there is a block (without loss of generality, assume it is a head block) of F containing more than one endpoint. Let h(i), h(j) be any two heads in this block. If t(i) is contained in arc j, then j is dominated by i. If t(j) is contained in arc i, then i is dominated by j. In any case, F contains a dominated arc, a contradiction.

Now, assume F is proper and each block of F contains exactly one endpoint. Let i be an arc of F. We show that for every arc j overlapping with i, there exists another arc u overlapping with i but not with j. Let $i_1$, ..., $i_k$ be the list of all arcs whose heads are contained in arc i and ordered clockwise from h(i) (see Figure 3). Let PREV($i_1$), ..., PREV($i_k$) be the list of all arcs whose tails are contained in arc i and ordered clockwise from h(i) (note that PREV($i_m$) could be the same as one of $i_1$, ..., $i_k$). Since there exists exactly one tail between any two heads, $i_t$, PREV($i_t$), t = 1, ..., k are all the arcs in F overlapping with i. If any $i_p$ equals any PREV($i_q$), then every arc passes through t(i) and F must be a clique, a contradiction.

Figure. 3 Those arcs whose heads are contained in arc i

Claim. Arc $i_t$ does not overlap with arc PREV($i_t$) for t = 1, ..., k.

Proof. Suppose $i_1$ overlaps with PREV($i_1$). Then we claim that F must be a clique. First of all, we show that i overlaps with PREV(i), namely, PREV(i) is among $i_1$, ..., $i_k$. Because F is proper, h(PREV(i)) must be contained in segment (t(PREV($i_1$)),h(PREV($i_1$))). Consider the arc j whose tail is contained in

$(h(PREV(i)),h(PREV(i_1))$ and is the most clockwise such tail before $h(PREV(i_1))$. Since F is proper, $h(i_1)$ must be in arc j. Since $t(i)$ is in arc j, this forces j to be arc i (otherwise, j would properly contain i). Hence, PREV(i) is among arcs $i_1, ..., i_k$ and the union of $\{i\}$, $\{i_1,...,i_k\}$ and $\{PREV(i_1),...,PREV(i_k)\}$ is the set of all arcs in F. Now, every arc in F except i passes through $h(PREV(i_1))$. Hence $F\backslash\{i\}$ is a clique. Since i overlaps with every arc in $F\backslash\{i\}$, F is also a clique.

**End of proof of Claim.** ▮

Hence, $i_1$ does not overlap with $PREV(i_1)$. Similar arguments can be used to show that $i_t$ does not overlap with $PREV(i_t)$ for $t = 2, ..., k$. ▮

A graph G is said to be (n,k)–*circular* if there exists an (circular) ordering of its vertices into $v_1, ..., v_n$ such that two vertices $v_i$ and $v_j$ with $i < j$ are adjacent iff either $j - i \leq k$ or $n - j + i \leq k$. G is called *circular* if it is (n,k)–circular for some integer k. Note that an (n,k)–circular graph G is a clique iff $n \leq 2k + 1$. For $n < j < 2n$, define $v_j = v_{j-n}$. If G is a clique, define $GD(v_i) = \{v_i\}$ for each $v_i$. Otherwise, for each $v_i$, define $GD(v_i) = \{v_i,v_{i+k+1},...,v_{i+mk+m}\}$, where $m = \lfloor n/k \rfloor$. If G is k–circular, then $GD(v_i)$ is a maximal independent set for each $v_i$.

**Theorem 2.** *If G is $(n,k)$–circular and $v_1, ..., v_n$ is a circular ordering, then $GD(v_i)$ is a maximum independent set in G for every $v_i$.*

Proof. For any two vertices $v_i$ and $v_j$ with $i < j$ the mapping $f : v_k \to v_{k+j-i}$ is an isomorphism for G. Hence, $|GD(v_i)| = |GD(v_j)|$. Now, let $v_i$ be a vertex contained in some MIS P of G. Then, by an argument similar to that of Lemma 2, we have $|P| = |GD(v_i)|$. □

Step 2 of our algorithm is trivial by the following

Lemma 4. *If F is strictly proper, then its corresponding circular–arc graph G is circular. Furthermore, the clockwise head ordering gives a circular ordering.*

Proof. Assume G is not a clique. Choose an arbitrary vertex of G as $v_1$. Then choose $v_2$, $v_3$, ..., $v_n$ according to the clockwise head order of their corresponding arcs starting from the head of the arc for $v_1$. Denote the corresponding arc of $v_i$ by $u_i$, $i = 1, ..., n$. Suppose, for some integer k, arcs $u_2$, ..., $u_{k+1}$ have their heads contained in $u_1$ but $u_{k+2}$ does not. Then we claim that G is (n,k)–circular. We shall show that each $v_i$ is adjacent to $v_{i+1}$, ..., $v_{i+k}$ but not to $v_{i+k+1}$ and hence, the clockwise head ordering gives a circular ordering.

By Claim 1 of Theorem 2, $u_2$ overlaps with neither PREV($u_2$) nor SUCC($u_2$). Because F is strictly proper, there must exist an arc whose head is contained in segment $(t(u_1),t(u_2))$. In particular, $h(u_{k+2})$ must be contained in $(t(u_1),t(u_2))$. Suppose $h(u_{k+3})$ is also in segment $(t(u_1),t(u_2))$. Then there must exist an arc i whose tail is contained in $(h(u_{k+2}),h(u_{k+3}))$. Because F is proper, h(i) must be contained in $(h(u_1),h(u_2))$, contradictory to the fact that $h(u_2)$ is the first head following $h(u_1)$ in the clockwise traversal. Therefore, $h(u_{k+2})$ is the only head contained in $(t(u_1),t(u_2))$; $v_2$ is adjacent to $v_3$, ..., $v_{k+2}$ but not to $v_{k+3}$. The rest follows by induction. □

## 3. An O(n) algorithm for finding a strictly proper subfamily F" of F

Our algorithm for finding a strictly proper subfamily consists of two parts:

I. Reduce F to a proper subfamily F' (a simple algorithm is described in [2]).

II. Remove additional dominated arcs in F' to ensure that each block contains exactly one endpoint and obtain, in O(n) time, a strictly proper family F" satisfying $\alpha(F) = \alpha(F")$.

swarp It is not difficult to reduce F directly to a strictly proper subfamily in O(n) time. The reason to break this procedure into two parts is to simplify notations.

Assume the given family F' is proper, we use the following algorithm to further reduce it to a strictly proper subfamily. Construct a doubly linked circular list L for all the 2n endpoints ordered clockwise. In the following procedure, we use array A(p) to indicate whether the endpoint p is in L (when A(p) = 1), array D(p) to indicate whether p should be deleted from L (when D(p) = 1) and array M(p) to indicate whether p has been encountered (when M(p) = 1). We use a stack S to temporarily store certain endpoints to be deleted from L. The reason not to delete these endpoints immediately is that whenever an endpoint is processed, the block containing that endpoint is processed simultaneously (procedure ELIMINATE (B)). We describe Part II in Figure 4 below.

```
begin

Initialize A(p) to be 1, D(p) to be 0 and M(p) to be 0 for every endpoint p;
while there exists an endpoint p in L with M(p) = 0 do
    pick any endpoint p in L with M(p) = 0; let B be the block containing p;
    M(p) ← 1; call ELIMINATE (B);
    while S ≠ ∅ do
        let q be the top element in S; let B be the block containing q;
        if A(q) = 0 then S ← S\{q};
        else call ELIMINATE (B);
    endwhile;
endwhile;
end;

Procedure ELIMINATE (B)
1.  Repeat steps 2, 3 and 4 until |B| ≤ 1.
2.  If B contains an endpoint d with D(d) = 0, then if B is a head (respectively,
    tail) block, keep the most counterclockwise (respectively, clockwise) endpoint
    r in B with D(r) = 0. Set B' = B\{r}. Set B = {r}.
3.  Otherwise, let B' = B, B = merger of the two blocks adjacent to B.
4.  Delete all endpoints in B' from L. If B is a head (respectively, tail) block,
    place their undeleted tails (respectively, heads) in S. Set the D-entries of
    these latter endpoints to be 1.
```

**Figure 4. Reducing a proper family to a strictly proper family**

The A–entry, D–entry and M–entry of each endpoint will be changed at most once throughout the algorithm. Furthermore, each endpoint can be placed in S at most once. Hence each endpoint is processed at most a constant number of times and the total running time of the algorithm is O(n).

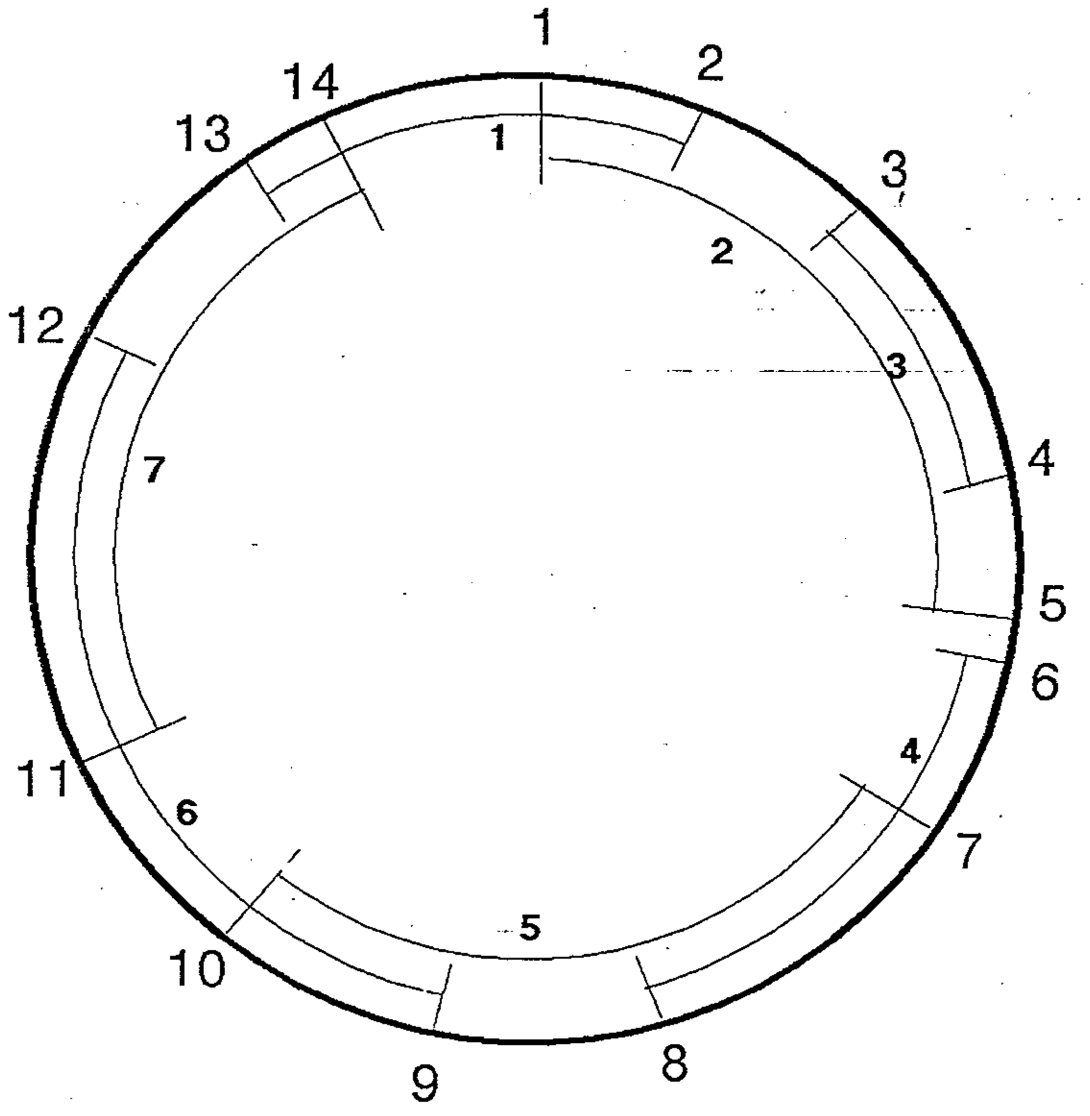
Correctness of the algorithm:

Each time a block is processed, its size is reduced to at most one (step 1 of the procedure ELIMINATE). At the end, each block contains exactly one endpoint. Hence, all we have to show is that only dominated arcs are deleted. If we encounter a block B which contains some arcs with zero D–entries, then because F is proper, all arcs with zero D–entries whose heads (respectively, tail) are contained in B except the one, say r, with the most counterclockwise (respectively, clockwise) endpoint are dominated by r and deleted in step 2 of the procedure ELIMINATE.

## References

1. U.I. Gupta, D.T. Lee and J. Y.–T. Leung, *Efficient algorithms for interval graphs and circular–arc graphs*, Networks, 12 (1982), pp. 459–467.

2. S. Masuda and K. Nakajima, *An optimal algorithm for finding a maximum independent set of a circular–arc graph*, **SIAM J. Comput.**, 17 (1988), pp. 41–52.



arc 1 = (13,2)

arc 2 = (1,5)

arc 3 = (3,4)

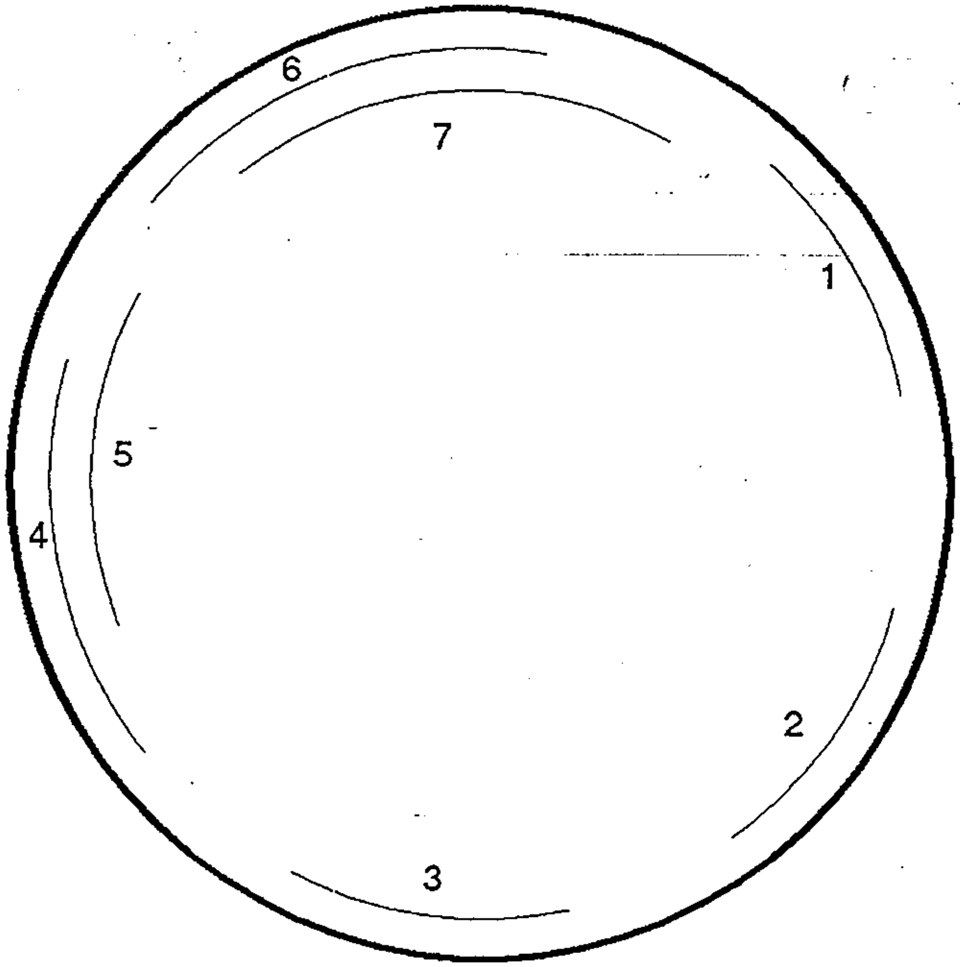
arc 4 = (6,8)

arc 5 = (7,10)

arc 6 = (9,12)

arc 7 = (11,14)

**Figure 1. A circular – arc family**



Two strictly proper subfamilies of F with the same MIS size

{ 1, 2, 3, 4, 6 } and { 1, 2, 3, 5, 7 }

**Figure 2. Two different strictly proper subfamilies**

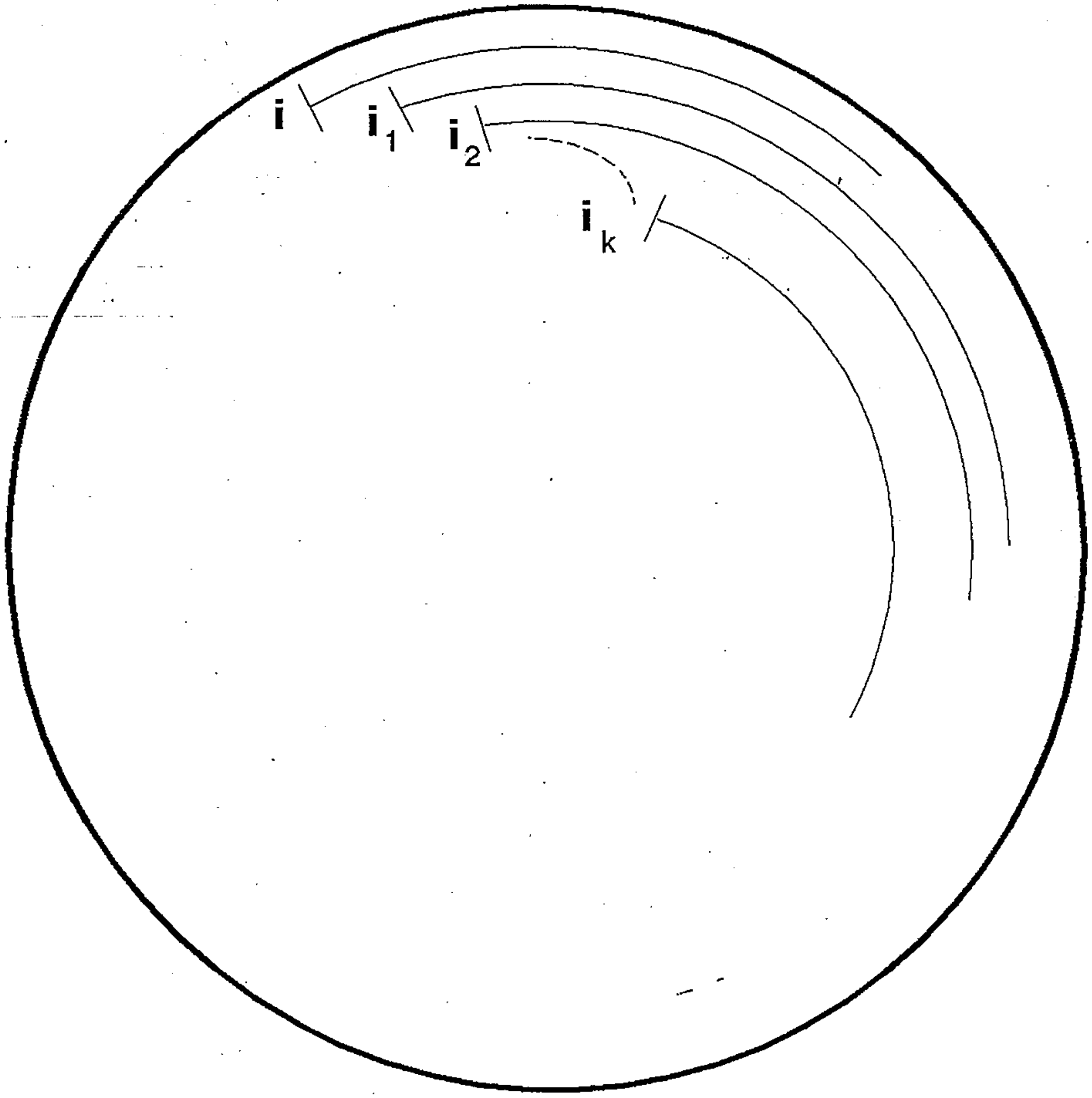


Figure 3. Those arcs whose heads are contained in arc i