The experiments listed below analyze the space complexity of the algorithm focusing on optimizing the k (no. of partial orders) and m (partial order set size) values.

Experiment 1:

Rationale: Examine the relation between k and m.

For a fixed n = 100, variations of k with respect to m.

(a) k <= n, m <= n

As k increases, m decreases.

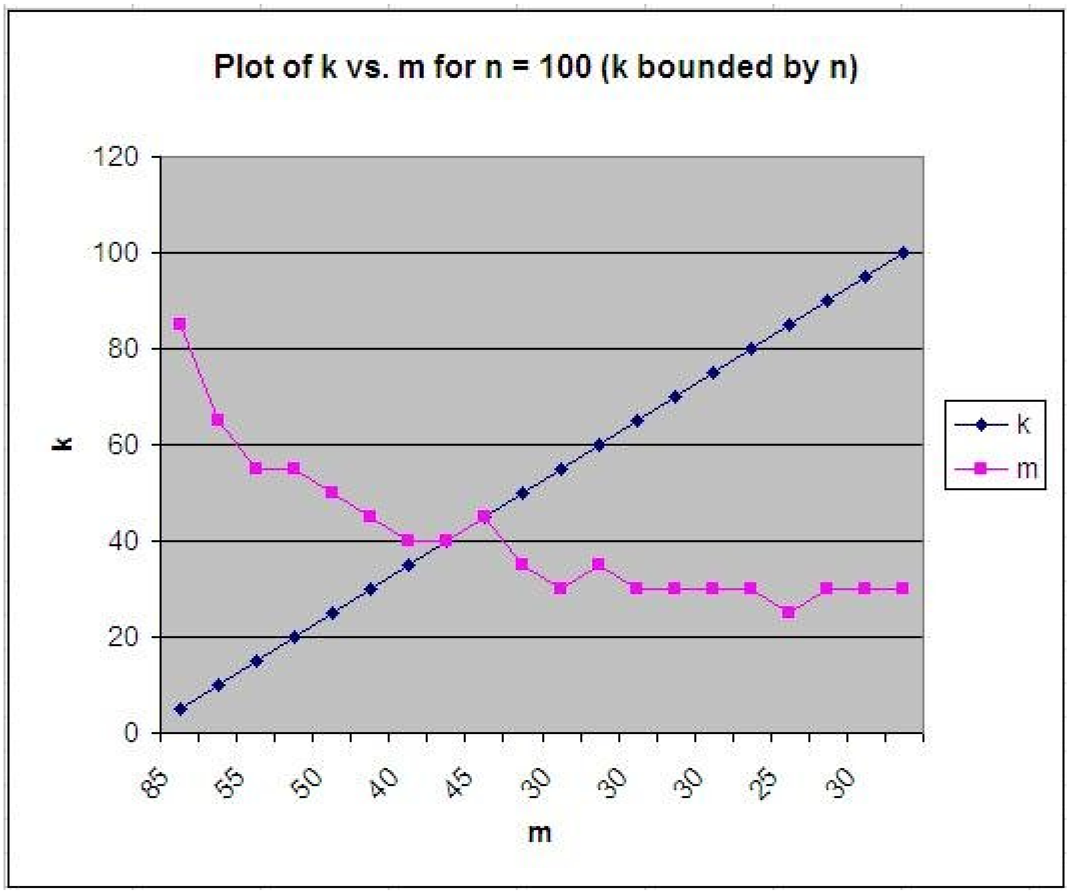

Table 1.1

k	m	k x m
5	85	425
10	65	650
15	55	825
20	55	1100
25	50	1250

30	45	1350
35	40	1400
40	40	1600
45	45	2025
50	35	1750
55	30	1650
60	35	2100
65	30	1950
70	30	2100
75	30	2250
80	30	2400
85	25	2125
90	30	2700
95	30	2850
100	30	3000

(b) k – unbounded, m <= n

As m increases, k decreases.

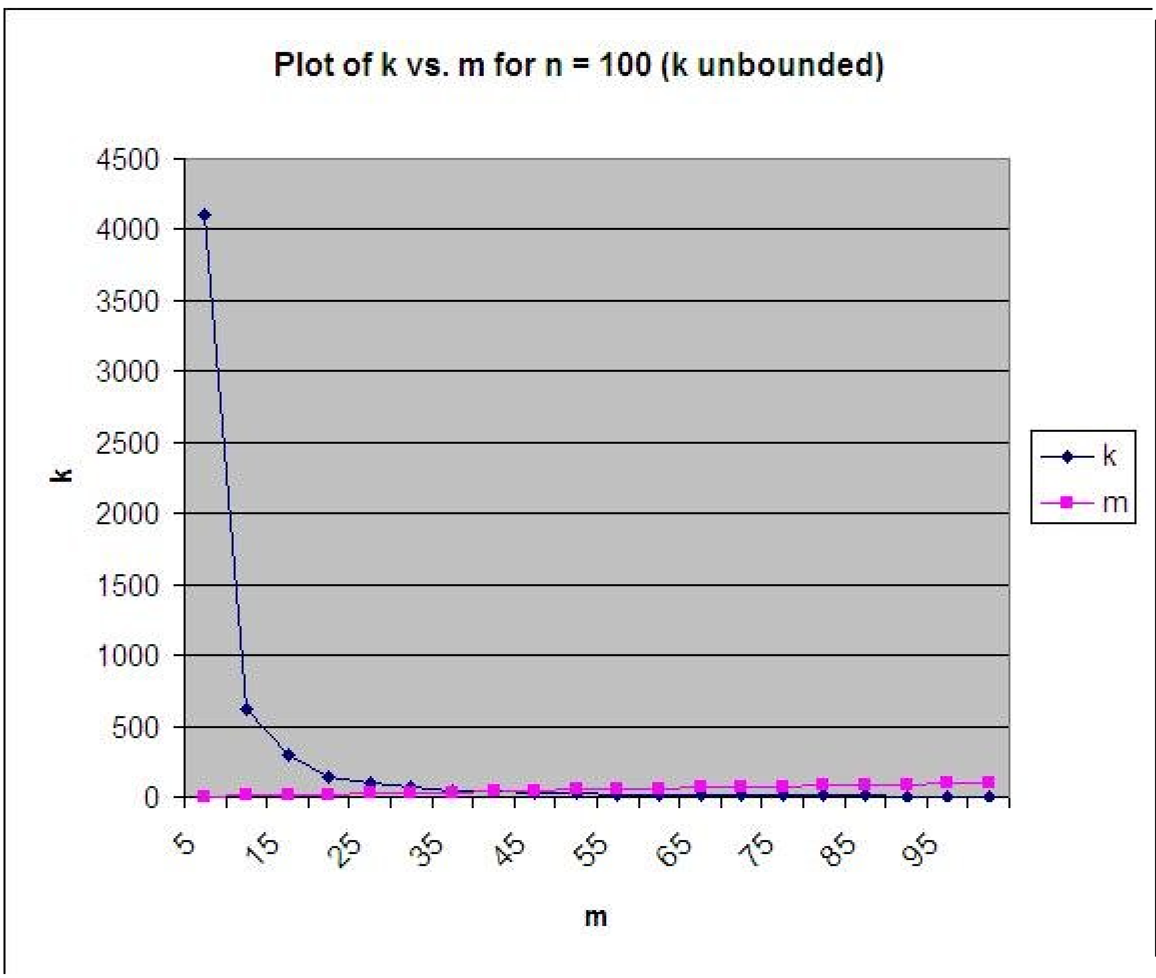

Table 1.2

k	**m**	**k x m**
4110	5	2055
615	10	6150
295	15	4425
135	20	2700
95	25	2375
70	30	2100
45	35	1575
40	40	1600
30	45	1350
25	50	1250
20	55	1100
20	60	1200
15	65	975
10	70	700
15	75	1125
10	80	800
10	85	850
5	90	450
5	95	475
5	100	500

Observations:

1. In both the above cases, k is inversely proportional to m. Smaller is the partial order size (m), greater is the number of people (k) required to order them to get back the correct total order.

2. The product of k and m (k x m) decides the space required to store the partial orders. This number must be minimized to optimize the space complexity. From the above two tables, we can see that the product k x m is minimum when m is closest to n. However, it is not practical to have partial orders (m) the size of the total number of elements (n).

Experiment 2:

Rationale: Examine the m for k as a multiple of n.

For different values of n (300 >= n >= 10), the partial order size is fixed at m = 10 and k is a multiple of n k = C x n (C is a constant.)

(a) k = n
(b) k = 2 x n
(c) k = 5 x n
(d) k = 10 x n
(e) k = 50 x n

The following table gives the flag value deciding whether we get back the total order depending on the number of partial orders as listed above.

Flag = 0 The total order is obtained from partial orders is correct.
Flag = 1 The total order is obtained from partial orders is incorrect.

Table 2.

n	**(a) Flag (k = n)**	**(b) Flag (k = 2 x n)**	**© Flag (k = 5 x n)**	**(d) Flag (k = 10 x n)**	**(e) Flag (k = 50 x n)**
10	1	1	1	1	1
20	1	1	1	1	1
30	0	1	1	1	1
40	0	0	1	1	1
50	0	0	1	1	1
60	0	0	0	1	1
70	0	0	1	1	1
80	0	0	0	1	1
90	0	0	0	1	1
100	0	0	0	1	1
110	0	0	0	1	1
120	0	0	0	1	1
130	0	0	0	0	1
140	0	0	0	0	1
150	0	0	0	0	1
160	0	0	0	0	1
170	0	0	0	0	1
180	0	0	0	0	1
190	0	0	0	0	1
200	0	0	0	0	1
210	0	0	0	0	1
220	0	0	0	0	1
230	0	0	0	0	1
240	0	0	0	0	1
250	0	0	0	0	1

260	0	0	0	0	1
270	0	0	0	0	1
280	0	0	0	0	1
290	0	0	0	0	1
300	0	0	0	0	0

Observations:

1. The first four cases, k = n, k = 2 x n, k = 5 x n and k = 10 x n give back the total orders when the total number of elements (n) is small. However, for larger values, such a k does not get back the total order.

2. The last case where k = 50 x n works for even larger values of n (e.g. n = 290). However, it does not give back the total order for n greater that 300.

3. So, in the function k = C x n, if C is a constant selected from the set of all positive integers, it will give back the correct total order for the cases where C is also derived from a function f(n).

Experiment 3 (I):

Rationale: Derive the functional dependency of k on n.

For different values of n (300 >= n >= 10), the partial order size is fixed at m = 10 and k is a multiple of n

k = f(n) x n

(a) f(n) = n   k = n x n
(b) f(n) = n/2   k = n/2 x n
(c) f(n) = sqrt(n)   k = sqrt(n) x n
(d) f(n) = log(n)   k = log(n) x n

The following table gives the flag value deciding whether we get back the total order depending on the number of partial orders as listed above.

Flag = 0   The total order is obtained from partial orders is correct.
Flag = 1   The total order is obtained from partial orders is incorrect.

Table 3.1

n	Flag (k = n x n)	Flag (k = n/2 x n)	Flag (k = sqrt(n) x n)	Flag (k = log(n) x n)
10	1	1	1	1
20	1	1	1	1
30	1	1	1	1
40	1	1	1	1
50	1	1	1	1
60	1	1	1	1
70	1	1	1	0
80	1	1	1	0
90	1	1	1	0
100	1	1	1	0
110	1	1	1	0
120	1	1	1	0
130	1	1	0	0
140	1	1	0	0
150	1	1	1	0
160	1	1	0	0
170	1	1	0	0
180	1	1	0	0
190	1	1	0	0
200	1	1	0	0
210	1	1	0	0
220	1	1	0	0
230	1	1	0	0
240	1	1	0	0
250	1	1	0	0
260	1	1	0	0
270	1	1	0	0
280	1	1	0	0
290	1	1	0	0
300	1	1	0	0

Observations:

1. The first two cases k = n x n, k = n x n/2 give back the total order correctly. On the other hand, k = sqrt(n) x n and k = log(n) x n yield very poor results, especially when n gets larger.

2. From these results we can deduce that k must be a function of n such that

$$k = f(n) \times n$$

and

$$f(n) = c \times n \quad \text{where } 0 < c <= 1$$

Experiment 3 (II):

Rationale: Derive the closest constant relating k with n^2.

For different values of n (300 >= n >= 10), the partial order size is fixed at m = 10 and k is a multiple of n

$$k = C \times n^2$$

(a) C = 1/4 $\quad$ k = n/4 x n
(b) C = 1/6 $\quad$ k = n/6 x n
(c) C = 1/8 $\quad$ k = n/8 x n

Table 3.2

n	**Flag (k = (1/4) x n^2)**	**Flag (k = (1/6) x n^2)**	**Flag (k = (1/8) x n^2)**
10	1	1	1
20	1	1	1
30	1	1	1
40	1	1	1
50	1	1	1
60	1	1	1
70	1	1	1
80	1	1	1
90	1	1	1
100	1	1	1
110	1	1	1
120	1	1	1
130	1	1	1
140	1	1	1
150	1	1	1
160	1	1	1
170	1	0	1
180	1	1	0
190	1	1	1

200	1	1	1
210	1	1	0
220	1	1	1
230	1	1	1
240	1	1	1
250	1	1	0
260	1	1	0
270	1	1	1
280	1	1	0
290	1	1	0
300	1	1	0
310	1	0	0
320	1	1	0
330	1	0	0
340	1	1	0
350	1	1	0
360	1	1	0
370	1	1	0
380	1	1	0
390	1	1	0
400	1	1	0

Observations:

1. The first case, $k = (1/4)\ n^2$, gives back a correct total order every single time. The second case $k = (1/6)\ n^2$, give back the total order correctly too most of the time. The third case, $(1/8)\ n^2$, does not give back the total order on many occasions.

2. From these results we can deduce that k must be a function of n such that

   $$k = (1/6)\ n^2$$

   which also includes $k = (1/4)\ n^2$. The constant C can be safely established to be approximately 0.1667 or 1/6.

Experiment 4:

Rationale: Derive the exact m value in the case where k = n.

For different values of n (300 >= n >= 10), the number of partial orders is fixed at n an the exact the partial order size is derived for such a k

$$k = n$$

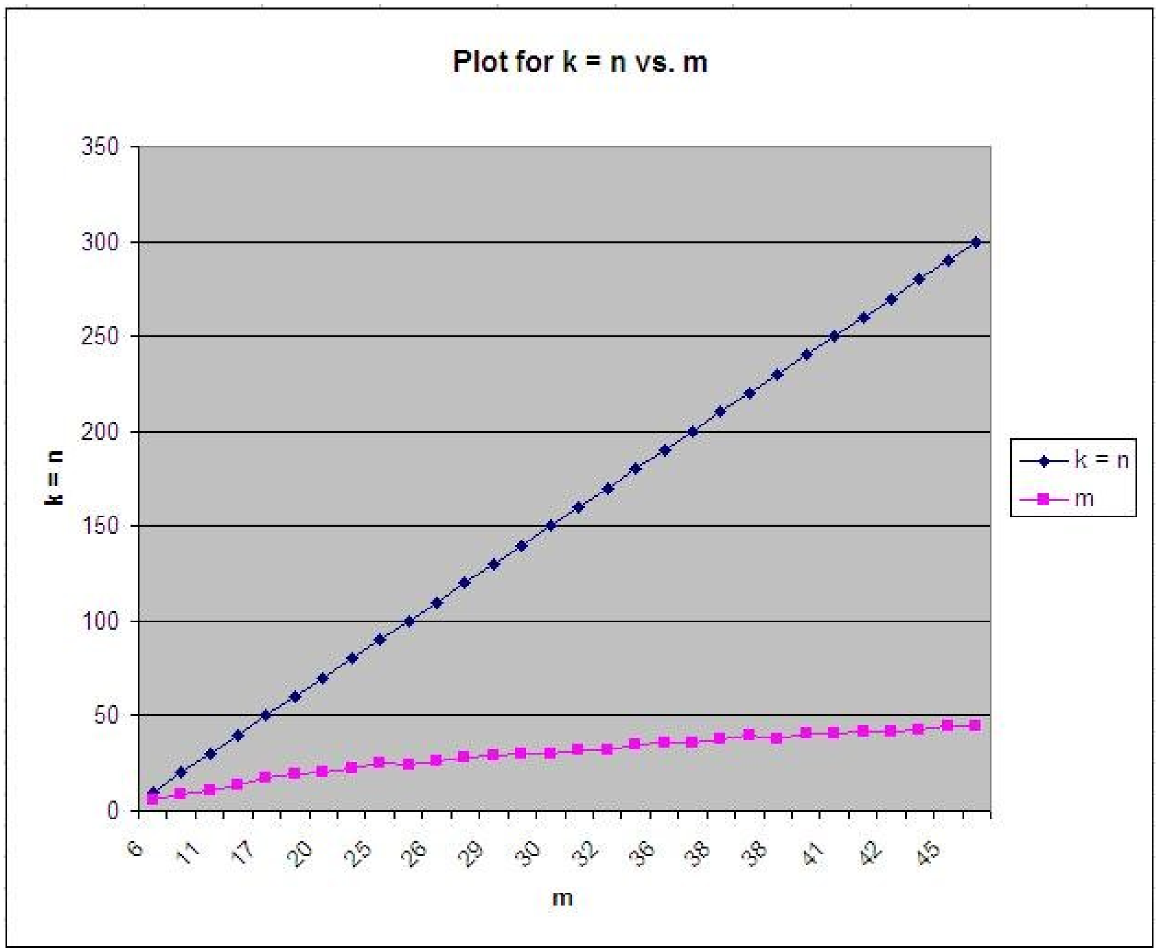

Table 4.

n	k	m	m/n x 100
10	10	6	60
20	20	9	45
30	30	11	36
40	40	14	35
50	50	17	34
60	60	19	31
70	70	20	28
80	80	22	27
90	90	25	27
100	100	24	24
110	110	26	23
120	120	28	23

130	130	29	22
140	140	30	21
150	150	30	20
160	160	32	20
170	170	32	18
180	180	35	19
190	190	36	18
200	200	36	18
210	210	38	18
220	220	40	18
230	230	38	16
240	240	41	17
250	250	41	16
260	260	42	16
270	270	42	15
280	280	43	15
290	290	45	15
300	300	45	15

Observations:

1. The partial order set size as a fraction of the total set size decreases as the n increases (also k increases with n).

2. The m value is closer to n when n is small (so is k).