

# 3<sup>rd</sup> N QUEENS ETSI Plugtests™ CONTEST

## Counting the number of solutions Single and Distributed Program



N-QUEENS-TESTCASE-2006-v1.doc patrick.guillemine@etsi.org - 21 Août 2006

<b>GOAL</b> .....	<b>1</b>
<b>WHAT IS THE N QUEENS PROBLEM?</b> .....	<b>2</b>
HOW MANY WAYS TO PLACE 8 QUEENS (CHESS) ON A 8X8 CHESSBOARD? .....	2
SHORT HISTORY OF N QUEENS PROBLEM .....	3
EXAMPLE OF USE .....	3
KNOW SOLUTIONS FOR N=4 TO 25 .....	3
<b>WHAT IS THE BEST CURRENT ALGORITHM?</b> .....	<b>4</b>
BEST PROGRAM FOR ONE MACHINE .....	4
PROGRAM DESIGNED FOR DISTRIBUTION .....	5
KNOWN MANUAL DISTRIBUTION OF THE N QUEENS PROBLEM .....	5
<i>Fixing the first column</i> .....	5
<i>Si(N) for N=8 to 20</i> .....	6
<i>Fixing first 2 and first 3 columns to count solutions for N=21 and N=22</i> .....	6
<b>MANUAL DISTRIBUTED COMPUTING</b> .....	<b>6</b>
HOW SOLUTIONS FOR N=21 WERE COMPUTED? .....	7
HOW SOLUTIONS FOR N=22 WERE COMPUTED ? .....	8
Si(N) FOR N=18 TO 22 .....	9
<b>WHY AND HOW USING GRID COMPUTING TO SOLVE THE PROBLEM WHEN N&gt;23 ?</b> .....	<b>10</b>
<b>APPENDIX : NQUEENS.C</b> .....	<b>10</b>
<b>APPENDIX : ND1.C</b> .....	<b>14</b>
<b>APPENDIX : ND12.C</b> .....	<b>15</b>
<b>APPENDIX : ND123.C</b> .....	<b>17</b>
<b>APPENDIX 199 RESULTS FOR N=21</b> .....	<b>19</b>
<b>APPENDIX : SLOANE A059963 UPDATE FOR N=1 TO 21</b> .....	<b>20</b>

### Goal

The goal of this document is to answer the following questions:

- What is the N Queens problem for us?
- What was the algorithm use at CNAM – University of Nice – France?
- Why and how to use GRID COMPUTING to solve the N-Queens counting problem when N>23?
- N Queens problem, a good test case for distributed computing?

## What is the N Queens problem?

To understand what this problem is, let us start with the origin, counting the 92 solutions to the 8 queens' problem.

### *How many ways to place 8 Queens (chess) on a 8x8 chessboard?*

The eight queens' puzzle is the problem of putting eight chess queens on an 8x8 chessboard such that none of them is able to capture any other using the standard chess queen's moves. That is to say, no two queens should share the same row, column, or diagonal. The generalised problem of placing n "non-dominating" queens on an n by n chessboard was posed as early as 1850.

The eight queens' problem has 92 solutions or 12 distinct solutions if symmetry operations such as rotations and reflections of the board are taken into consideration.

1 5 8 6 3 7 2 4	3 6 8 1 5 7 2 4	5 1 4 6 8 2 7 3	6 3 1 8 5 2 4 7
1 6 8 3 7 4 2 5	3 6 8 2 4 1 7 5	5 1 8 4 2 7 3 6	6 3 5 7 1 4 2 8
1 7 4 6 8 2 5 3	3 7 2 8 5 1 4 6	5 1 8 6 3 7 2 4	6 3 5 8 1 4 2 7
1 7 5 8 2 4 6 3	3 7 2 8 6 4 1 5	5 2 4 6 8 3 1 7	6 3 7 2 4 8 1 5
2 4 6 8 3 1 7 5	3 8 4 7 1 6 2 5	5 2 4 7 3 8 6 1	6 3 7 2 8 5 1 4
2 5 7 1 3 8 6 4	4 1 5 8 2 7 3 6	5 2 6 1 7 4 8 3	6 3 7 4 1 8 2 5
2 5 7 4 1 8 6 3	4 1 5 8 6 3 7 2	5 2 8 1 4 7 3 6	6 4 1 5 8 2 7 3
2 6 1 7 4 8 3 5	4 2 5 8 6 1 3 7	5 3 1 6 8 2 4 7	6 4 2 8 5 7 1 3
2 6 8 3 1 4 7 5	4 2 7 3 6 8 1 5	5 3 1 7 2 8 6 4	6 4 7 1 3 5 2 8
2 7 3 6 8 5 1 4	4 2 7 3 6 8 5 1	5 3 8 4 7 1 6 2	6 4 7 1 8 2 5 3
2 7 5 8 1 4 6 3	4 2 7 5 1 8 6 3	5 7 1 3 8 6 4 2	6 8 2 4 1 7 5 3
2 8 6 1 3 5 7 4	4 2 8 5 7 1 3 6	5 7 1 4 2 8 6 3	7 1 3 8 6 4 2 5
3 1 7 5 8 2 4 6	4 2 8 6 1 3 5 7	5 7 2 4 8 1 3 6	7 2 4 1 8 5 3 6
3 5 2 8 1 7 4 6	4 6 1 5 2 8 3 7	5 7 2 6 3 1 4 8	7 2 6 3 1 4 8 5
3 5 2 8 6 4 7 1	4 6 8 2 7 1 3 5	5 7 2 6 3 1 8 4	7 3 1 6 8 5 2 4
3 5 7 1 4 2 8 6	4 6 8 3 1 7 5 2	5 7 4 1 3 8 6 2	7 3 8 2 5 1 6 4
3 5 8 4 1 7 2 6	4 7 1 8 5 2 6 3	5 8 4 1 3 6 2 7	7 4 2 5 8 1 3 6
3 6 2 5 8 1 7 4	4 7 3 8 2 5 1 6	5 8 4 1 7 2 6 3	7 4 2 8 6 1 3 5
3 6 2 7 1 4 8 5	4 7 5 2 6 1 3 8	6 1 5 2 8 3 7 4	7 5 3 1 6 8 2 4
3 6 2 7 5 1 8 4	4 7 5 3 1 6 8 2	6 2 7 1 3 5 8 4	8 2 4 1 7 5 3 6
3 6 4 1 8 5 7 2	4 8 1 3 6 2 7 5	6 2 7 1 4 8 5 3	8 2 5 3 1 7 4 6
3 6 4 2 8 5 7 1	4 8 1 5 7 2 6 3	6 3 1 7 5 8 2 4	8 3 1 6 2 5 7 4
3 6 8 1 4 7 5 2	4 8 5 3 1 7 2 6	6 3 1 8 4 2 7 5	8 4 1 3 6 2 7 5

92 SOLUTIONS

The coding system can be explained by an example

		*					
					*		
			*				
	*						*
				*			
						*	
*							
1	5	8	6	3	7	2	4

This problem is well known as "EIGHT QUEENS PUZZLE" with a lot of hits in Internet search engines. Notice that the 92 solutions are a subset of all 8! (40 320) permutations of 1 2 3 4 5 6 7 8

### **Short History of $n$ queens problem**

The  $n$ -queens problem is an old puzzle that has been around for more than a century. Originally known as the 8-Queens problem, it has been studied by many famous mathematicians over the years, including the great German mathematician Karl Friedrich Gauss (1777-1855). Franz Nauck generalised the problem to  $n$  by  $n$  boards in 1850. Since the 1960's, with rapid developments in computer science, this problem has been used as an example of backtracking algorithms, permutation generation, divide and conquer paradigm, program development methodology, constraint satisfaction problems, integer programming, and specification.

---

This problem is now a standard in algorithm design of software engineering. The rumour saying that Gauss did not find all the 92 solutions is a hoax.

### **Example of use**

The queens problem is really a puzzle but, surprisingly, there are some practical applications such as parallel memory storage schemes, VLSI testing, traffic control, and deadlock prevention.

On the Internet we can find algorithm finding **one** solution for high value of  $N$  computed in minimum time. This is another aspect of this problem, for the moment we did not try to involve constraint satisfaction problem technique in the search (counting, enumerating) of the total number of solutions to the  $n$  queens' problem. This option is still opened, if interested look at: <http://www.cit.gu.edu.au/~sotic/nqueens.html>  
*"The  $n$ -queens problem has been studied in relationship to the research on the constraint satisfaction problem. The goal of the research is to develop fast and practical solutions to large scale constraint satisfaction problems."*

In our case, we **count** the **total number** of all possible solutions without storing or displaying it. The elapsed time to display all the solutions is quite high for  $N > 18$ . The storage space required to store all the solutions (even if we use high compression techniques) is quite big too. This problem exponentially grows in complexity.

### **Know solutions for $N=4$ to 25**

N	Solutions	Record owner	Ref.
4	2		
5	10		
6	4		
7	40		
8	92		
9	352		
10	724		
11	2,680		
12	14,200		
13	73,712		
14	365,596		
15	2,279,184		
16	14,772,512		
17	95,815,104		
18	666,090,624		
19	4,968,057,848		
20	39,029,188,884		
21	314,666,222,712		
22	2,691,008,701,644		
23	24,233,937,684,440	INRIA	
24	227,514,171,973,736	Takaken	
25	2,207,893,435,808,352	INRIA	<a href="http://proactive.objectweb.org">http://proactive.objectweb.org</a>

On the Internet, there is a place to store this records, called by us SLOANE 's collection: On-Line Encyclopedia of Integer Sequences <http://www.research.att.com/~njas/sequences/Seis.html>

The known and verified solutions of the N queens (N<25) problem are available at:  
<http://www.research.att.com/cgi-bin/access.cgi/as/njas/sequences/eisA.cgi?Anum=000170>

\*\*\* continue here \*\*\*

## What is the best current algorithm?

We do not pretend to know the best program ever used. In the different experiences we lived with this problem, we discovered how important is the power of the CPU (100 MHz, 1GHz, 2GHz) and the efficiency of the algorithm. The design of the program is obviously not the same on a single machine or for a high number of computers.

### **Best program for one machine**

To produce the SLOANE results with a program running on one machine, the current best **open source** program we found is <http://www.ic-net.or.jp/home/takaken/e/queen/> (\*) using both Sylvain PION (INRIA France, record holder for N=23) bit wise technique and all symmetry optimizations. It can count solutions for N<=18 in less than 5 minutes.

(\*) nqueens.c downloaded from this location is available in appendix

<----- N-Queens Solutions ----->		<----- time ----->	
N: Total	Unique	days	hh:mm:ss.--
2: 0	0		0.00
3: 0	0		0.00
4: 2	1		0.00
5: 10	2		0.00
6: 4	1		0.00
7: 40	6		0.00
8: 92	12		0.00
9: 352	46		0.00
10: 724	92		0.00
11: 2680	341		0.00
12: 14200	1787		0.00
13: 73712	9233		0.06
14: 365596	45752		0.16
15: 2279184	285053		0.88
16: 14772512	1846955		5.93
17: 95815104	11977939		40.76
18: 666090624	83263591		4:55.50
19: 4968057848	621012754		37:49.96
20: 39029188884	4878666808		5:03:35.16
21: 314666222712	39333324973	1	18:40:00.09

CPU = Athlon XP 2100+ (1.73GHz)  
Compiler = Visual C++ 5.0  
1 Day 18 hours 40 mn for 21 Queens! WOAW

Depending on the power of the algorithm and the CPU of your computer, you will be able to count the number of solutions using a single program on a single machine in less than hours, maximum one week for N<20. At one stage or another, you will need to use many computers to count the total number of solution in less than one hour, day or week!

The best mathematical approach I know is provided by Yuh-Pyng (Arping) Shieh [arping@turing.csie.ntu.edu.tw](mailto:arping@turing.csie.ntu.edu.tw) <http://goedel.csie.ntu.edu.tw/~arping/turing/cm/index.htm>  
Arping gives us copies of PhDs and bibliography and executable version of his program.

**Program designed for distribution**

For  $N \geq 20$  you need to cut the big calculation into small pieces of independent programs running in parallel on many computers.

The program able to split the problem into many calculations is not the same as the best program designed for one machine (nqueens.c).

For some students the bit wise approach (like in nqueens.c) was not obvious enough to divide the problem, they preferred to use less optimised but more understandable program. So we provide nd1.c in appendix. Usage is "**nd1.exe <Number\_Of\_Queens> <Position\_Of\_First\_Column>**". Example of usage is given in tables hereafter. I like to call this problem the Challenge Question "**N C1**"

The rule for distributing is:

- > Use non optimized (but more understandable) program to distribute the calculations
- Use optimized program to calculate/count the number of solutions for N with x queens fixed in positions (lines) : Challenge Question "**N C1 C2 C3 ... Cx**"

**Known Manual Distribution of the N queens problem**

Now we will see how the program can be distributed in many calculations where first columns are fixed.

**Fixing the first column**

For  $N=8$ , in the 92 solutions there are

Number of Solutions	Beginning with	Naming convention	Usage of nd1.exe	Challenge Question
4	1	S1	Nd1.exe 8 1	" <b>8 1</b> "
8	2	S2	Nd1.exe 8 2	" <b>8 2</b> "
16	3	S3	Nd1.exe 8 3	" <b>8 3</b> "
18	4	S4	Nd1.exe 8 4	" <b>8 4</b> "
18	5	S5	Nd1.exe 8 5	" <b>8 5</b> "
16	6	S6	Nd1.exe 8 6	" <b>8 6</b> "
8	7	S7	Nd1.exe 8 7	" <b>8 7</b> "
4	8	S8	Nd1.exe 8 8	" <b>8 8</b> "

Let us notice the horizontal symmetry where  $S1=S8, S2=S7, S3=S6$  and  $S4=S5$  (when N is even) This can be generalized when N is even. In this case the total amount of solutions is  $92=2*(S1+S2+S3+S4)$

When  $N=9$  we have  $S1=S9, S2=S8, S3=S7, S4=S6$  and  $S5$  is alone (When N is odd)

Number of Solutions	Beginning with	Naming convention	Usage of nd1.exe	Challenge Question
28	1	S1	Nd1.exe 9 1	" <b>9 1</b> "
30	2	S2	Nd1.exe 9 2	" <b>9 2</b> "
47	3	S3	Nd1.exe 9 3	" <b>9 3</b> "
44	4	S4	Nd1.exe 9 4	" <b>9 4</b> "
54	5	S5	Nd1.exe 9 5	" <b>9 5</b> "
44	6	S6	Nd1.exe 9 6	" <b>9 6</b> "
47	7	S7	Nd1.exe 9 7	" <b>9 7</b> "
30	8	S8	Nd1.exe 9 8	" <b>9 8</b> "
28	9	S9	Nd1.exe 9 9	" <b>9 9</b> "

In this case the total amount of solutions is  $352=S5+2*(S1+S2+S3+S4)$

In SLOANE collection there is a serie counting these values  $S1(N), S2(N), S3(N), \dots S21(N)$  for  $N=1$  to 21 <http://www.research.att.com/cgi-bin/access.cgi/as/njas/sequences/eisA.cgi?Anum=A059963>

On a single computer with nd1.c (in appendix) it was possible to compute  $S1, S2, \dots S20$  in less than one week. This was computed in 2002 without bit wise technique.

For N from 8 to 20, giving all values of Si(N) in less than one week could be a first pass of local or distributed calculation in a GRID COMPUTING test event. Expected results for Si(N) with N=8 to 20 are

Si(N) for N=8 to 20

N	8	9	10	11	12	13	14	15	16	17	18	19	20
S1	4	28	64	96	500	2 760	11 892	69 516	436 228	2 729 772	17 210 372	121 956 044	912 695 924
S2	8	30	48	219	806	3 799	16 488	98 156	569 531	3 321 745	22 038 667	154 458 256	1 134 501 243
S3	16	47	65	209	1 165	5 508	23 024	122 763	736 363	4 423 207	27 585 497	187 854 702	1 381 017 109
S4	18	44	93	295	1 359	6 023	27 494	157 034	892 999	5 172 708	33 297 967	230 334 612	1 649 528 539
S5		54	92	346	1 631	7 346	32 163	175 296	1 050 762	6 214 709	39 005 536	263 322 762	1 925 960 786
S6				350	1 639	7 385	34 760	201 164	1 160 280	6 787 111	43 698 287	300 359 104	2 158 815 033
S7						8 070	36 977	206 294	1 249 262	7 546 991	47 802 996	328 342 530	2 388 912 956
S8								218 738	1 290 831	7 698 195	50 523 766	351 816 544	2 563 311 029
S9										8 026 228	51 882 224	360 352 268	2 670 502 723
S10												370 464 204	2 729 349 100

Fixing first 2 and first 3 columns to count solutions for N=21 and N=22

With friends, colleagues and students attending « Software Engineering » (Génie Logiciel CNAM/GL) and « Test and Validation Techniques » (Techniques de Tests et Validations du Logiciel CNAM/TV) in French University of Nice where I am teaching, we computed S1(N), S2(N), S3(N), ... ,S11(N) for N=21 and N=22 in, respectively, less than 2 and 3 weeks. The distribution was done using (bit wise without all symmetry properties) nd12.c and nd123.c (in appendix). The assignation and consolidation of each individual calculation was done “manually” by email, using excel cut/paste of text log files. The goal was to get a first full computation results and to show at least one example of the distribution mechanism.

## Manual distributed computing

In 2001/2002, I gave the N queens counting problem for the first time as exercise in CNAM (Nice University – France) « Tests and Validation techniques ». The challenge was to count solution up to N=20 in less than one week. This was done with one computer (CPU 1GHz), running an algorithm without bit wise approach in less than one week. Some students implemented C#, rpc and Winsock programs in Perl, Java, C++ and C languages.

In 2003/2004 in CNAM, I gave the previous results to new students and asked again to solve and improve the N queens problem, and the challenge was to approach the world record N=23. Because I had only a record of the total number of solutions and because I wanted to show students how calculations could be distributed, I did the distribution of work and consolidation manually (by email)

**How solutions for N=21 were computed?**

I gave nd12.c, nd12.exe and cygwin1.dll to students and assigned them 199 calculations (21 C1 C2) with C1 and C2 in the following list:

C1	C2	C1	C2	C1	C2	C1	C2	C1	C2	C1	C2	C1	C2	C1	C2	C1	C2				
1	3	2	4	3	1	4	1	5	1	6	1	7	1	8	1	9	1	10	1	11	1
1	4	2	5	3	5	4	2	5	2	6	2	7	2	8	2	9	2	10	2	11	2
1	5	2	6	3	6	4	6	5	3	6	3	7	3	8	3	9	3	10	3	11	3
1	6	2	7	3	7	4	7	5	7	6	4	7	4	8	4	9	4	10	4	11	4
1	7	2	8	3	8	4	8	5	8	6	8	7	5	8	5	9	5	10	5	11	5
1	8	2	9	3	9	4	9	5	9	6	9	7	9	8	6	9	6	10	6	11	6
1	9	2	10	3	10	4	10	5	10	6	10	7	10	8	10	9	7	10	7	11	7
1	10	2	11	3	11	4	11	5	11	6	11	7	11	8	11	9	11	10	8	11	8
1	11	2	12	3	12	4	12	5	12	6	12	7	12	8	12	9	12	10	12	11	9
1	12	2	13	3	13	4	13	5	13	6	13	7	13	8	13	9	13	10	13	11	13
1	13	2	14	3	14	4	14	5	14	6	14	7	14	8	14	9	14	10	14	11	14
1	14	2	15	3	15	4	15	5	15	6	15	7	15	8	15	9	15	10	15	11	15
1	15	2	16	3	16	4	16	5	16	6	16	7	16	8	16	9	16	10	16	11	16
1	16	2	17	3	17	4	17	5	17	6	17	7	17	8	17	9	17	10	17	11	17
1	17	2	18	3	18	4	18	5	18	6	18	7	18	8	18	9	18	10	18	11	18
1	18	2	19	3	19	4	19	5	19	6	19	7	19	8	19	9	19	10	19	11	19
1	19	2	20	3	20	4	20	5	20	6	20	7	20	8	20	9	20	10	20	11	20
1	20	2	21	3	21	4	21	5	21	6	21	7	21	8	21	9	21	10	21	11	21
1	21																				

```
Each participant ran a Windows 2000 & XP batch (ex: "go21-10-3.bat" ) containing
date /t >> 21-10-3.log
time /t >> 21-10-3.log
nd12.exe 21 10 3 >> 21-10-3.log
date /t >> 21-10-3.log
time /t >> 21-10-3.log
exit
```

I call this the Challenge Question "21 10 3"

I consolidated the 199 calculations (in appendix) to find  
 Challenge Question

S1(21)	=	7 063 988 992	"21 1"
S2(21)	=	8 479 630 929	"21 2"
S3(21)	=	10 449 374 683	"21 3"
S4(21)	=	12 239 272 958	"21 4"
S5(21)	=	14 388 837 943	"21 5"
S6(21)	=	16 034 294 995	"21 6"
S7(21)	=	17 934 352 665	"21 7"
S8(21)	=	19 124 084 606	"21 8"
S9(21)	=	20 363 051 828	"21 9"
S10(21)	=	20 727 857 478	"21 10"
S11(21)	=	21 056 728 558	"21 11"

And we can check that Sylain Pion value (with odd symmetrical property) is matching  
 $314\ 666\ 222\ 712 = S11 + 2*(S1 + S2 + S3 + S4 + S5 + S6 + S7 + S8 + S9 + S10)$

Both the 21 sub totals and all the 199 results can be used as test cases !,  
 For details see "Appendix 199 results for N=21"

**How solutions for N=22 were computed ?**

I gave nd123.c, nd123.exe and cygwin1.dll to students and assigned them 210 batch calculations; a set of all patterns 22 C1 C2 C3 (all Challenge Questions "22 C1 C2 C3") with the same beginning 22 C1 C2. We can use the notation Challenge Question "22 \* \* \*" The total number of individual calculations is 3458 distributed in the 210 batches

22-1-10.bat	22-11-22.bat	22-4-13.bat	22-6-4.bat	22-9-19.bat
22-1-11.bat	22-11-3.bat	22-4-14.bat	22-6-8.bat	22-9-2.bat
22-1-12.bat	22-11-4.bat	22-4-15.bat	22-6-9.bat	22-9-20.bat
22-1-13.bat	22-11-5.bat	22-4-16.bat	22-7-1.bat	22-9-21.bat
22-1-14.bat	22-11-6.bat	22-4-17.bat	22-7-10.bat	22-9-22.bat
22-1-15.bat	22-11-7.bat	22-4-18.bat	22-7-11.bat	22-9-3.bat
22-1-16.bat	22-11-8.bat	22-4-19.bat	22-7-12.bat	22-9-4.bat
22-1-17.bat	22-11-9.bat	22-4-2.bat	22-7-13.bat	22-9-5.bat
22-1-18.bat	22-2-10.bat	22-4-20.bat	22-7-14.bat	22-9-6.bat
22-1-19.bat	22-2-11.bat	22-4-21.bat	22-7-15.bat	22-9-7.bat
22-1-20.bat	22-2-12.bat	22-4-22.bat	22-7-16.bat	
22-1-21.bat	22-2-13.bat	22-4-6.bat	22-7-17.bat	210 batches
22-1-22.bat	22-2-14.bat	22-4-7.bat	22-7-18.bat	
22-1-3.bat	22-2-15.bat	22-4-8.bat	22-7-19.bat	
22-1-4.bat	22-2-16.bat	22-4-9.bat	22-7-2.bat	
22-1-5.bat	22-2-17.bat	22-5-1.bat	22-7-20.bat	
22-1-6.bat	22-2-18.bat	22-5-10.bat	22-7-21.bat	
22-1-7.bat	22-2-19.bat	22-5-11.bat	22-7-22.bat	
22-1-8.bat	22-2-20.bat	22-5-12.bat	22-7-3.bat	
22-1-9.bat	22-2-21.bat	22-5-13.bat	22-7-4.bat	
22-10-1.bat	22-2-22.bat	22-5-14.bat	22-7-5.bat	
22-10-12.bat	22-2-4.bat	22-5-15.bat	22-7-9.bat	
22-10-13.bat	22-2-5.bat	22-5-16.bat	22-8-1.bat	
22-10-14.bat	22-2-6.bat	22-5-17.bat	22-8-10.bat	
22-10-15.bat	22-2-7.bat	22-5-18.bat	22-8-11.bat	
22-10-16.bat	22-2-8.bat	22-5-19.bat	22-8-12.bat	
22-10-17.bat	22-2-9.bat	22-5-2.bat	22-8-13.bat	
22-10-18.bat	22-3-1.bat	22-5-20.bat	22-8-14.bat	
22-10-19.bat	22-3-10.bat	22-5-21.bat	22-8-15.bat	
22-10-2.bat	22-3-11.bat	22-5-22.bat	22-8-16.bat	
22-10-20.bat	22-3-12.bat	22-5-3.bat	22-8-17.bat	
22-10-21.bat	22-3-13.bat	22-5-7.bat	22-8-18.bat	
22-10-22.bat	22-3-14.bat	22-5-8.bat	22-8-19.bat	
22-10-3.bat	22-3-15.bat	22-5-9.bat	22-8-2.bat	
22-10-4.bat	22-3-16.bat	22-6-1.bat	22-8-20.bat	
22-10-5.bat	22-3-17.bat	22-6-10.bat	22-8-21.bat	
22-10-6.bat	22-3-18.bat	22-6-11.bat	22-8-22.bat	
22-10-7.bat	22-3-19.bat	22-6-12.bat	22-8-3.bat	
22-10-8.bat	22-3-20.bat	22-6-13.bat	22-8-4.bat	
22-11-1.bat	22-3-21.bat	22-6-14.bat	22-8-5.bat	
22-11-13.bat	22-3-22.bat	22-6-15.bat	22-8-6.bat	
22-11-14.bat	22-3-5.bat	22-6-16.bat	22-9-1.bat	
22-11-15.bat	22-3-6.bat	22-6-17.bat	22-9-11.bat	
22-11-16.bat	22-3-7.bat	22-6-18.bat	22-9-12.bat	
22-11-17.bat	22-3-8.bat	22-6-19.bat	22-9-13.bat	
22-11-18.bat	22-3-9.bat	22-6-2.bat	22-9-14.bat	
22-11-19.bat	22-4-1.bat	22-6-20.bat	22-9-15.bat	
22-11-2.bat	22-4-10.bat	22-6-21.bat	22-9-16.bat	
22-11-20.bat	22-4-11.bat	22-6-22.bat	22-9-17.bat	
22-11-21.bat	22-4-12.bat	22-6-3.bat	22-9-18.bat	

Windows 2000 & XP batch "**22-1-10.bat**" content is

```
start "1-10-2" /B /LOW /WAIT nd123.exe 22 1 10 2 >> all22.log
start "1-10-4" /B /LOW /WAIT nd123.exe 22 1 10 4 >> all22.log
start "1-10-5" /B /LOW /WAIT nd123.exe 22 1 10 5 >> all22.log
start "1-10-6" /B /LOW /WAIT nd123.exe 22 1 10 6 >> all22.log
start "1-10-7" /B /LOW /WAIT nd123.exe 22 1 10 7 >> all22.log
start "1-10-8" /B /LOW /WAIT nd123.exe 22 1 10 8 >> all22.log
start "1-10-12" /B /LOW /WAIT nd123.exe 22 1 10 12 >> all22.log
start "1-10-13" /B /LOW /WAIT nd123.exe 22 1 10 13 >> all22.log
start "1-10-14" /B /LOW /WAIT nd123.exe 22 1 10 14 >> all22.log
start "1-10-15" /B /LOW /WAIT nd123.exe 22 1 10 15 >> all22.log
start "1-10-16" /B /LOW /WAIT nd123.exe 22 1 10 16 >> all22.log
start "1-10-17" /B /LOW /WAIT nd123.exe 22 1 10 17 >> all22.log
start "1-10-18" /B /LOW /WAIT nd123.exe 22 1 10 18 >> all22.log
start "1-10-19" /B /LOW /WAIT nd123.exe 22 1 10 19 >> all22.log
start "1-10-20" /B /LOW /WAIT nd123.exe 22 1 10 20 >> all22.log
start "1-10-21" /B /LOW /WAIT nd123.exe 22 1 10 21 >> all22.log
start "1-10-22" /B /LOW /WAIT nd123.exe 22 1 10 22 >> all22.log
```

3458 results for N=22 have been done in **4547h 31mn** of CPU time. The computation lasted **439h 51mn** between 18 Jan and 5 Feb 2004, see file "**22Queens-Si.xls**"

		Challenge	Question
S1(22)	= 56 624 964 700	"22 1"	
S2(22)	= 69 021 492 236	"22 2"	
S3(22)	= 83 035 719 395	"22 3"	
S4(22)	= 97 778 677 945	"22 4"	
S5(22)	= 113 530 277 161	"22 5"	
S6(22)	= 128 055 647 633	"22 6"	
S7(22)	= 141 701 163 201	"22 7"	
S8(22)	= 153 417 424 175	"22 8"	
S9(22)	= 162 641 629 340	"22 9"	
S10(22)	= 168 619 397 689	"22 10"	
S11(22)	= 171 077 957 347	"22 11"	

And we can check that Sylain Pion value (with even symmetrical property) is matching  $2\ 691\ 008\ 701\ 644 = 2 * (S1+S2+S3+S4+S5+S6+S7+S8+S9+S10+S11)$

**Si(N) for N=18 to 22**

Si(N) are the same than we I call Challenge Question "**N i**"

N	18	19	20	21	22
S1	17 210 372	121 956 044	912 695 924	7 063 988 992	56 624 964 700
S2	22 038 667	154 458 256	1 134 501 243	8 479 630 929	69 021 492 236
S3	27 585 497	187 854 702	1 381 017 109	10 449 374 683	83 035 719 395
S4	33 297 967	230 334 612	1 649 528 539	12 239 272 958	97 778 677 945
S5	39 005 536	263 322 762	1 925 960 786	14 388 837 943	113 530 277 161
S6	43 698 287	300 359 104	2 158 815 033	16 034 294 995	128 055 647 633
S7	47 802 996	328 342 530	2 388 912 956	17 934 352 665	141 701 163 201
S8	50 523 766	351 816 544	2 563 311 029	19 124 084 606	153 417 424 175
S9	51 882 224	360 352 268	2 670 502 723	20 363 051 828	162 641 629 340
S10		370 464 204	2 729 349 100	20 727 857 478	168 619 397 689
S11				21 056 728 558	171 077 957 347

Record published at

<http://www.research.att.com/cgi-bin/access.cgi/as/njas/sequences/eisA.cgi?Anum=A059963>

## Why and how using GRID COMPUTING to solve the problem when N>23 ?

Use GRID COMPUTING, Use Proactive ! <http://www-sop.inria.fr/oasis/proactive/>  
And register to the N-queens counting problem challenge in 2 phases (  
<http://www.etsi.org/plugtests/GRID.htm> )

- Remote and preliminary qualification challenge
- Face-to-face challenge on 19 October 2004 at ETSI Plugtests Sophia Antipolis France

Described in document N-QUEENS-COUNTING-CHALLENGE-v3.doc you can obtain by sending a mail to [Patrick.guillemin@etsi.org](mailto:Patrick.guillemin@etsi.org) Cc [plugtests@etsi.org](mailto:plugtests@etsi.org)

### Appendix : nqueens.c

```
/* ***** */
/* N-Queens Solutions ver3.1          takaken July/2003          */
/* ***** */
#include <stdio.h>
#include <time.h>

#define MAXSIZE 24
#define MINSIZE 2

int SIZE, SIZEE;
int BOARD[MAXSIZE], *BOARDE, *BOARD1, *BOARD2;
int MASK, TOPBIT, SIDEMASK, LASTMASK, ENDBIT;
int BOUND1, BOUND2;

__int64 COUNT8, COUNT4, COUNT2;
__int64 TOTAL, UNIQUE;

/* ***** */
/* Display the Board Image          */
/* ***** */
void Display(void)
{
    int y, bit;

    printf("N= %d\n", SIZE);
    for (y=0; y<SIZE; y++) {
        for (bit=TOPBIT; bit; bit>>=1)
            printf("%s ", (BOARD[y] & bit)? "Q": "-");
        printf("\n");
    }
    printf("\n");
}
/* ***** */
/* Check Unique Solutions          */
/* ***** */
void Check(void)
{
    int *own, *you, bit, ptn;

    /* 90-degree rotation */
    if (*BOARD2 == 1) {
        for (ptn=2,own=BOARD+1; own<=BOARDE; own++,ptn<=1) {
            bit = 1;
```

```
        for (you=BOARDE; *you!=ptn && *own>=bit; you--)  
            bit <= 1;  
        if (*own > bit) return;  
        if (*own < bit) break;  
    }  
    if (own > BOARDE) {  
        COUNT2++;  
        //Display();  
        return;  
    }  
}  
  
/* 180-degree rotation */  
if (*BOARDE == ENDBIT) {  
    for (you=BOARDE-1,own=BOARD+1; own<=BOARDE; own++,you--) {  
        bit = 1;  
        for (ptn=TOPBIT; ptn!=*you && *own>=bit; ptn>>=1)  
            bit <= 1;  
        if (*own > bit) return;  
        if (*own < bit) break;  
    }  
    if (own > BOARDE) {  
        COUNT4++;  
        //Display();  
        return;  
    }  
}  
  
/* 270-degree rotation */  
if (*BOARD1 == TOPBIT) {  
    for (ptn=TOPBIT>>1,own=BOARD+1; own<=BOARDE; own++,ptn>>=1) {  
        bit = 1;  
        for (you=BOARD; *you!=ptn && *own>=bit; you++)  
            bit <= 1;  
        if (*own > bit) return;  
        if (*own < bit) break;  
    }  
}  
COUNT8++;  
//Display();  
}  
/*****/  
/* First queen is inside */  
/*****/  
void Backtrack2(int y, int left, int down, int right)  
{  
    int bitmap, bit;  
  
    bitmap = MASK & ~(left | down | right);  
    if (y == SIZEE) {  
        if (bitmap) {  
            if (!(bitmap & LASTMASK)) {  
                BOARD[y] = bitmap;  
                Check();  
            }  
        }  
    } else {  
        if (y < BOUND1) {  
            bitmap |= SIDEMASK;  
            bitmap ^= SIDEMASK;  
        }  
    }  
}
```

```

    } else if (y == BOUND2) {
        if (!(down & SIDEMASK)) return;
        if ((down & SIDEMASK) != SIDEMASK) bitmap &= SIDEMASK;
    }
    while (bitmap) {
        bitmap ^= BOARD[y] = bit = -bitmap & bitmap;
        Backtrack2(y+1, (left | bit)<<1, down | bit, (right | bit)>>1);
    }
}
/*****
/* First queen is in the corner */
*****/
void Backtrack1(int y, int left, int down, int right)
{
    int bitmap, bit;

    bitmap = MASK & ~(left | down | right);
    if (y == SIZEE) {
        if (bitmap) {
            BOARD[y] = bitmap;
            COUNT8++;
            //Display();
        }
    } else {
        if (y < BOUND1) {
            bitmap |= 2;
            bitmap ^= 2;
        }
        while (bitmap) {
            bitmap ^= BOARD[y] = bit = -bitmap & bitmap;
            Backtrack1(y+1, (left | bit)<<1, down | bit, (right | bit)>>1);
        }
    }
}
/*****
/* Search of N-Queens */
*****/
void NQueens(void)
{
    int bit;

    /* Initialize */
    COUNT8 = COUNT4 = COUNT2 = 0;
    SIZEE = SIZE - 1;
    BOARDE = &BOARD[SIZEE];
    TOPBIT = 1 << SIZEE;
    MASK = (1 << SIZE) - 1;

    /* 0:000000001 */
    /* 1:011111100 */
    BOARD[0] = 1;
    for (BOUND1=2; BOUND1<SIZEE; BOUND1++) {
        BOARD[1] = bit = 1 << BOUND1;
        Backtrack1(2, (2 | bit)<<1, 1 | bit, bit>>1);
    }

    /* 0:000001110 */
    SIDEMASK = LASTMASK = TOPBIT | 1;
    ENDBIT = TOPBIT >> 1;

```

```

    for (BOUND1=1,BOUND2=SIZE-2; BOUND1<BOUND2; BOUND1++,BOUND2--) {
        BOARD1 = &BOARD[BOUND1];
        BOARD2 = &BOARD[BOUND2];
        BOARD[0] = bit = 1 << BOUND1;
        Backtrack2(1, bit<<1, bit, bit>>1);
        LASTMASK |= LASTMASK>>1 | LASTMASK<<1;
        ENDBIT >>= 1;
    }

    /* Unique and Total Solutions */
    UNIQUE = COUNT8      + COUNT4      + COUNT2;
    TOTAL  = COUNT8 * 8 + COUNT4 * 4 + COUNT2 * 2;
}
/*****
/* Format of Used Time
*****/
void TimeFormat(clock_t utime, char *form)
{
    int  dd, hh, mm;
    float ftime, ss;

    ftime = (float)utime / CLOCKS_PER_SEC;

    mm = (int)ftime / 60;
    ss = ftime - (float)(mm * 60);
    dd = mm / (24 * 60);
    mm = mm % (24 * 60);
    hh = mm / 60;
    mm = mm % 60;

    if (dd) sprintf(form, "%4d %02d:%02d:%05.2f", dd, hh, mm, ss);
    else if (hh) sprintf(form, "      %2d:%02d:%05.2f", hh, mm, ss);
    else if (mm) sprintf(form, "          %2d:%05.2f", mm, ss);
    else sprintf(form, "              %5.2f", ss);
}
/*****
/* N-Queens Solutions MAIN
*****/
int main(void)
{
    clock_t starttime;
    char form[20];

    printf("<----- N-Queens Solutions -----> <---- time ---->\n");
    printf(" N:          Total          Unique days hh:mm:ss.--\n");
    for (SIZE=MINSIZE; SIZE<=MAXSIZE; SIZE++) {
        starttime = clock();
        NQueens();
        TimeFormat(clock() - starttime, form);
        printf("%2d:%16I64d%16I64d %s\n", SIZE, TOTAL, UNIQUE, form);
    }

    return 0;
}

```

## Appendix : ND1.C

```
/* ND1.c Patrick 16 Jan 2002, L1 fixed*/
/* Usage nd1.exe N Lig1 */
#include <stdio.h>
#define maxN 32
#define dmaxN 64
#define vrai 1
#define faux 0
static int tableau [maxN];
static int libre [maxN];
static int okmont [dmaxN];
static int okdesc [dmaxN];
static unsigned long int total;
static int N;
int Lig1;
void permut (int);
void initialise (void);
void affiche (void);

main(argc, argv)
int argc; char *argv[];
{
if (argc==3)
{
N=atoi(argv[1]);
Lig1=atoi(argv[2]);
}
else
{
printf("Nb reines= ");scanf("%d",&N);
printf("Lig1= ");scanf("%d",&Lig1);
}
printf("Nb reines= %d\n",N);
printf("Lig1= %d\n",Lig1);
total=0;
initialise();
tableau[Lig1]=Lig1;
okmont[Lig1-1+N-1]=faux;
okdesc[Lig1+1-2]=faux;
libre[Lig1]=faux;
permut(2);
printf("Total= %u\n",total);
}

void initialise (void)
{
static int J;
for(J=1;J<=N;J++)
{
libre[J]=vrai;
}
for(J=0;J<=2*N;J++)
{
okmont[J]=vrai;
}
for(J=0;J<=N*2;J++)
{
okdesc[J]=vrai;
}
}

void permut (position)
int position;
{
```

```

int I;
if(position<=N)
{
    for(I=1;I<=N;I++)
    {
        if(libre[I]==vrai&&okmont[I-position+N-1]==vrai&&okdesc[I+position
2]==vrai)
        {
            tableau[position]=I;
            okmont[I-position+N-1]=faux;
            okdesc[I+position-2]=faux;
            libre[I]=faux;
            permut(position+1);
            tableau[position]=0;
            libre[I]=vrai;
            okmont[I-position+N-1]=vrai;
            okdesc[I+position-2]=vrai;
        }
    }
}
else
{
    total=total+1;
}
}

```

## Appendix : ND12.C

```

//*****\\
//    nd12.c Probleme des nreines    n-queens counting pb    \\
//    Version 1.0.0.4    M GELAS 2001, P GUILLEMIN 2004    \\
//    correction de l'overflow sur les solution pour n=19    \\
//    TRAVAIL EN COURS DANS CYGWIN BIN    \\
//    Patrick GUILLEMIN 10 Jan 2004    \\
//*****\\
#include "stdio.h"
#include "time.h"
#include "curses.h"
int i;
long dec;
int N;
int Lig1;
int Lig2;
long dec;
long xarrayoccupe, xarraydiagonalemont, xarraydiagonaledec;
long x1arrayoccupe, x1arraydiagonalemont, x1arraydiagonaledec;
long x2arrayoccupe, x2arraydiagonalemont, x2arraydiagonaledec;

static unsigned long int nbsolutionFinal;
void lpermutation(long , long ,long ,long ,long );

int main(int argc, char* argv[])
{
    if (argc==4)
    {
        N=atoi(argv[1]);
        Lig1=atoi(argv[2]);
        Lig2=atoi(argv[3]);
    }
    else
    {
        printf("Nb reines= ");scanf("%d",&N);
        printf("Lig1= ");scanf("%d",&Lig1);
        printf("Lig2= ");scanf("%d",&Lig2);
    }
}

```

```

printf("Nb reines= %d\n",N);
printf("Lig1= %d\n",Lig1);
printf("Lig2= %d\n",Lig2);

//Attention en interne Lig1=4 pour la ligne 5
Lig1=Lig1-1;
Lig2=Lig2-1;
nbsolutionFinal = 0;

// Mise en place du marquage pour Lig1

x1arrayoccupe = 1;
for (i=1;i<=Lig1;i++) {x1arrayoccupe = x1arrayoccupe << 1;}
x1arraydiagonalemont= (x1arrayoccupe << 1);
x1arraydiagonaledec = (x1arrayoccupe >> 1);

x2arrayoccupe = 1;
for (i=1;i<=Lig2;i++) {x2arrayoccupe = x2arrayoccupe << 1;}

xarraydiagonalemont = ((x1arraydiagonalemont | x2arrayoccupe) << 1) ;
xarraydiagonaledec = ((x1arraydiagonaledec | x2arrayoccupe) >> 1) ;
xarrayoccupe = x1arrayoccupe | x2arrayoccupe;

lpermutation(N, 3, xarraydiagonalemont, xarraydiagonaledec, xarrayoccupe);

printf("Total-bitwise = %u\n",nbsolutionFinal);
}

void lpermutation(
long n,
long position,
long parraydiagonalemont,
long parraydiagonaledec,
long parrayoccupe)
{
    long index,
        lposition,
        larrayoccupe,
        ivaleurdep,
        larraysolution,
        lenvironement;
    if (position>n) {
        nbsolutionFinal++;
    }
    else{
        lposition=position+1;
        larrayoccupe=1;
        lenvironement=parrayoccupe |
            parraydiagonalemont |
            parraydiagonaledec;
        for (index=1;index<=n;index++)
        {
            if ((larrayoccupe & lenvironement )==0)
            {
                lpermutation(n,
                    lposition,
                    ((parraydiagonalemont | larrayoccupe) << 1),
                    ((parraydiagonaledec | larrayoccupe) >> 1),
                    (parrayoccupe | larrayoccupe));
            }
            larrayoccupe=larrayoccupe << 1;
        }
    }
}
}

```

## Appendix : ND123.C

```
//*****\\
//      nd123.c Probleme des nreines  n-queens counting pb      \\
//      Version 1.0.0.5    M GELAS 2001, P GUILLEMIN 2004      \\
//      Patrick GUILLEMIN 11/1/04                               \\
//      modification pour -> nd123.exe                          \\
//      attention en interne les Lignes vont de 0 à n-1        \\
//*****\\
#include "stdio.h"
#include "time.h"
#include "curses.h"
int i;
long dec;
int N;
int Lig1;
int Lig2;
int Lig3;
long dec;
time_t debut;
time_t fin;
long xarrayoccupe, xarraydiagonalemont, xarraydiagonaledec;
long x1arrayoccupe, x1arraydiagonalemont, x1arraydiagonaledec;
long x2arrayoccupe, x2arraydiagonalemont, x2arraydiagonaledec;
long x3arrayoccupe, x3arraydiagonalemont, x3arraydiagonaledec;

static unsigned long int nbsolutionFinal;
void lpermutation(long , long ,long ,long ,long );

int main(int argc, char* argv[])
{
if (argc==5)
{
N=atoi(argv[1]);
Lig1=atoi(argv[2]);
Lig2=atoi(argv[3]);
Lig3=atoi(argv[4]);

}
else
{
printf("Nb reines= ");scanf("%d",&N);
printf("Lig1= ");scanf("%d",&Lig1);
printf("Lig2= ");scanf("%d",&Lig2);
printf("Lig3= ");scanf("%d",&Lig3);
}

printf("%d;",N);
printf("%d;",Lig1);
printf("%d;",Lig2);
printf("%d;",Lig3);

//Attention en interne Lig1=4 pour la ligne 5
Lig1=Lig1-1;
Lig2=Lig2-1;
Lig3=Lig3-1;

nbsolutionFinal = 0;

// Mise en place du marquage pour Lig1 Lig2 et Lig3

x1arrayoccupe = 1;
for (i=1;i<=Lig1;i++) {x1arrayoccupe = x1arrayoccupe << 1;}
x1arraydiagonalemont= (x1arrayoccupe << 1);
x1arraydiagonaledec = (x1arrayoccupe >> 1);

x2arrayoccupe = 1;
for (i=1;i<=Lig2;i++) {x2arrayoccupe = x2arrayoccupe << 1;}
```

```
x2arraydiagonalemont = ((x1arraydiagonalemont | x2arrayoccupe) << 1) ;
x2arraydiagonaledec  = ((x1arraydiagonaledec  | x2arrayoccupe) >> 1) ;
x2arrayoccupe        = x1arrayoccupe | x2arrayoccupe;

x3arrayoccupe = 1;
for (i=1;i<=Lig3;i++) {x3arrayoccupe = x3arrayoccupe << 1;}

xarraydiagonalemont = ((x2arraydiagonalemont | x3arrayoccupe) << 1) ;
xarraydiagonaledec  = ((x2arraydiagonaledec  | x3arrayoccupe) >> 1) ;
xarrayoccupe        = x2arrayoccupe | x3arrayoccupe;

time(&debut);

lpermutation(N, 4, xarraydiagonalemont, xarraydiagonaledec, xarrayoccupe);

printf("%u;",nbsolutionFinal);
time(&fin);

printf("%d\n",fin-debut);
}

void lpermutation(
long n,
long position,
long parraydiagonalemont,
long parraydiagonaledec,
long parrayoccupe)
{
    long index,
        lposition,
        larrayoccupe,
        ivaleurdep,
        larraysolution,
        lenvironement;
    if (position>n) {
        nbsolutionFinal++;
    }
    else{
        lposition=position+1;
        larrayoccupe=1;
        lenvironement=parrayoccupe |
            parraydiagonalemont |
            parraydiagonaledec;
        for (index=1;index<=n;index++)
        {
            if ((larrayoccupe & lenvironement )==0)
            {
                lpermutation(n,
                    lposition,
                    ((parraydiagonalemont | larrayoccupe) << 1),
                    ((parraydiagonaledec | larrayoccupe) >> 1),
                    (parrayoccupe | larrayoccupe));
            }
            larrayoccupe=larrayoccupe << 1;
        }
    }
}
}
```

### Appendix 199 results for N=21

21	1	3	190 441 438
21	1	4	232 247 043
21	1	5	292 372 661
21	1	6	351 009 352
21	1	7	394 692 776
21	1	8	448 959 069
21	1	9	474 303 516
21	1	10	514 915 146
21	1	11	504 048 511
21	1	12	532 841 022
21	1	13	493 433 582
21	1	14	505 596 239
21	1	15	440 497 440
21	1	16	424 112 047
21	1	17	349 160 755
21	1	18	320 913 768
21	1	19	239 405 759
21	1	20	207 875 370
21	1	21	147 163 498

21	2	4	300 739 787
21	2	5	351 190 625
21	2	6	429 781 115
21	2	7	494 621 010
21	2	8	538 974 661
21	2	9	584 911 396
21	2	10	607 887 723
21	2	11	636 431 187
21	2	12	621 196 809
21	2	13	633 276 670
21	2	14	582 187 218
21	2	15	575 008 246
21	2	16	496 710 431
21	2	17	463 592 111
21	2	18	376 829 475
21	2	19	332 531 766
21	2	20	246 311 609
21	2	21	207 449 090

21	3	1	198 693 110
21	3	5	455 406 879
21	3	6	537 347 363
21	3	7	612 142 254
21	3	8	692 692 476
21	3	9	713 720 014
21	3	10	770 955 656
21	3	11	763 698 214
21	3	12	798 449 604
21	3	13	747 395 297
21	3	14	760 200 544
21	3	15	669 930 162
21	3	16	654 975 388
21	3	17	549 551 244
21	3	18	512 549 136
21	3	19	401 679 913
21	3	20	354 438 232
21	3	21	255 549 197

21	4	1	243 034 421
21	4	2	313 395 153
21	4	6	646 026 949
21	4	7	729 284 151
21	4	8	804 987 125
21	4	9	879 832 778
21	4	10	888 660 142
21	4	11	931 198 451
21	4	12	910 871 631
21	4	13	922 911 239
21	4	14	856 901 474
21	4	15	837 299 475
21	4	16	733 167 022
21	4	17	691 540 219
21	4	18	577 055 163
21	4	19	519 265 443
21	4	20	404 784 308
21	4	21	349 057 814

21	5	1	336 755 692
21	5	2	392 126 023
21	5	3	481 133 921
21	5	7	880 000 529
21	5	8	983 998 800
21	5	9	1 012 123 843
21	5	10	1 104 873 386
21	5	11	1 070 686 833
21	5	12	1 120 392 585
21	5	13	1 049 911 743
21	5	14	1 054 249 022
21	5	15	946 310 391
21	5	16	916 717 964
21	5	17	779 028 088
21	5	18	734 160 847
21	5	19	589 386 345
21	5	20	531 244 746
21	5	21	405 737 185

21	6	1	392 060 003
21	6	2	491 803 336
21	6	3	574 254 491
21	6	4	684 928 489
21	6	8	1 100 735 109
21	6	9	1 188 142 437
21	6	10	1 196 340 494
21	6	11	1 254 243 926
21	6	12	1 204 830 063
21	6	13	1 224 607 582
21	6	14	1 135 322 390
21	6	15	1 110 248 534
21	6	16	984 143 988
21	6	17	921 288 775
21	6	18	784 131 454
21	6	19	714 287 706
21	6	20	571 442 450
21	6	21	501 483 768

21	7	1	481 007 786
21	7	2	582 080 422
21	7	3	700 703 726
21	7	4	810 607 123
21	7	5	924 690 858
21	7	9	1 315 457 307
21	7	10	1 399 807 293
21	7	11	1 361 961 838
21	7	12	1 425 315 637
21	7	13	1 313 571 564
21	7	14	1 320 172 894
21	7	15	1 183 102 791
21	7	16	1 160 933 751
21	7	17	1 004 233 645
21	7	18	940 934 861
21	7	19	764 423 376
21	7	20	696 561 838
21	7	21	548 785 955

21	8	1	549 395 191
21	8	2	646 774 886
21	8	3	782 834 421
21	8	4	913 208 978
21	8	5	1 034 000 589
21	8	6	1 143 365 488
21	8	10	1 456 853 837
21	8	11	1 500 835 498
21	8	12	1 451 182 720
21	8	13	1 470 400 285
21	8	14	1 359 635 062
21	8	15	1 323 931 983
21	8	16	1 183 426 289
21	8	17	1 132 709 835
21	8	18	967 866 087
21	8	19	868 040 366
21	8	20	705 357 553
21	8	21	634 265 538

21	9	1	605 803 372
21	9	2	737 157 567
21	9	3	844 017 586
21	9	4	1 019 353 173
21	9	5	1 125 230 796
21	9	6	1 275 723 776
21	9	7	1 355 827 868
21	9	11	1 547 962 987
21	9	12	1 583 647 618
21	9	13	1 501 284 337
21	9	14	1 520 445 341
21	9	15	1 371 406 973
21	9	16	1 325 035 640
21	9	17	1 151 229 624
21	9	18	1 073 644 276
21	9	19	891 664 153
21	9	20	792 932 922
21	9	21	640 683 819

21	10	1	648 884 104
21	10	2	744 190 970
21	10	3	904 353 261
21	10	4	1 020 503 835
21	10	5	1 200 400 698
21	10	6	1 292 946 277
21	10	7	1 424 575 279
21	10	8	1 481 658 888
21	10	12	1 555 957 521
21	10	13	1 558 459 523
21	10	14	1 489 115 330
21	10	15	1 459 094 002
21	10	16	1 301 701 314
21	10	17	1 222 501 908
21	10	18	1 047 073 324
21	10	19	931 956 958
21	10	20	770 416 376
21	10	21	674 067 910

21	11	1	659 753 476
21	11	2	800 708 159
21	11	3	902 888 087
21	11	4	1 099 040 811
21	11	5	1 178 192 890
21	11	6	1 369 864 589
21	11	7	1 424 951 794
21	11	8	1 548 746 696
21	11	9	1 544 217 777
21	11	13	1 544 217 777
21	11	14	1 548 746 696
21	11	15	1 424 951 794
21	11	16	1 369 864 589
21	11	17	1 178 192 890
21	11	18	1 099 040 811
21	11	19	902 888 087
21	11	20	800 708 159
21	11	21	659 753 476

## Appendix : SLOANE A059963 Update for N=1 to 21

De: Patrick René Guillemin

Date: ven. 16/01/2004 21:57 (checked on 20 February 2004)

à: 'njas(AT)research.att.com'

Cc: Alexandre.Di\_Costanzo(AT)sophia.inria.fr; arlette.bermond(AT)laposte.net; Benhaim06(AT)aol.com; BenoitR(AT)Experian-Scorex.com; cauve\_alexis(AT)yahoo.fr; ccardon(AT)amadeus.net; didier.gromaire(AT)laposte.net; eddyfice(AT)tele2.fr; francoisandrieu(AT)maxigrames.com; fred.ben(AT)wanadoo.fr; gacoch(AT)ifrance.com; isuttle(AT)amadeus.net; J.GERARDIN(AT)monaco-telecom.mc; J.LHUIILLIER(AT)monaco-telecom.mc; jdecroix(AT)ifrance.com; jduriez(AT)amadeus.net; Jean-Luc Freisse; Jeremy Rudel; J.L.OLIVIERO(AT)monaco-telecom.mc; larabi222(AT)yahoo.fr; Laurent Vreck; lefevre(AT)evolutionmm.com; leroy\_cyrill(AT)yahoo.fr; MatheodeNice(AT)aol.com; maxence.louasse(AT)libertysurf.fr; Niem Lu; Norbert Maurin; o.maj(AT)free.fr; o.paillery(AT)free.fr; pbouillon(AT)mis.mc; Sebastien.FLAMANT(AT)obs-nice.fr; v\_gontcharov(AT)hotmail.com; Yves Coueque; souchon.sebastien(AT)wanadoo.fr; patrickchevaux(AT)wanadoo.fr; Laurent.Toutain(AT)irisa.fr; damien(AT)RadioCockpit.com; philippe.henri(AT)9online.fr; P.Raffaelli(AT)sbm.mc; guillemin.jean-pierre(AT)wanadoo.fr; franck.lapira(AT)generale-des-eaux.net; lhardyj(AT)wanadoo.fr; michael.koch(AT)acanthis.fr; seb.lhote(AT)laposte.net  
Objet: RE : Re A059963

Dear All,

Dear Neil Sloane,

Here we are ! We (CNAM, colleagues and Friends) found together the new values of A059963 (\*) in the collection for N=21. This represents 1600 hours of calculations done in 214 hours of elapsed time. The calculations were divided in 199 sub-calculations.

(\*) <http://www.research.att.com/cgi-bin/access.cgi/as/njas/sequences/eisA.cgi?Anum=A059963>

The new sequence for A059963 (for N=1 to N=21) is

1,  
0,0,  
0,0,0,  
0,1,1,0,  
2,2,2,2,2,  
0,1,1,1,1,0,  
4,7,6,6,6,7,4,  
4,8,16,18,18,16,8,4,  
28,30,47,44,54,44,47,30,28,  
64,48,65,93,92,92,93,65,48,64,  
96,219,209,295,346,350,346,295,209,219,96,  
500,806,1165,1359,1631,1639,1639,1631,1359,1165,806,500,  
2760,3799,5508,6023,7346,7385,8070,7385,7346,6023,5508,3799,2760,  
11892,16488,23024,27494,32163,34760,36977,36977,34760,32163,27494,23024,16488,11892,  
69516,98156,122763,157034,175296,201164,206294,218738,206294,201164,175296,157034,122763,98156,69516,  
436228,569531,736363,892999,1050762,1160280,1249262,1290831,1290831,1249262,1160280,1050762,892999,736363,569531,436228,  
2729772,3321745,4423207,5172708,6214709,6787111,7546991,7698195,8026228,7698195,7546991,6787111,6214709,5172708,4423207,3321745,2729772,  
17210372,22038667,27585497,33297967,39005536,43698287,47802996,50523766,51882224,51882224,50523766,47802996,43698287,39005536,33297967,27585497,22038667,17210372,  
121956044,154458256,187854702,230334612,263322762,300359104,328342530,351816544,360352268,370464204,360352268,351816544,328342530,300359104,263322762,230334612,187854702,154458256,121956044,  
912695924,1134501243,1381017109,1649528539,1925960786,2158815033,2388912956,2563311029,2670502723,2729349100,2729349100,2670502723,2563311029,2388912956,2158815033,1925960786,1649528539,1381017109,1134501243,912695924,  
7063988992,8479630929,10449374683,12239272958,14388837943,16034294995,17934352665,19124084606,20363051828,20727857478,21056728558,20727857478,20363051828,19124084606,17934352665,16034294995,14388837943,12239272958,10449374683,8479630929,7063988992

The proof for N=21 is here :

S1 = 7 063 988 992  
S2 = 8 479 630 929  
S3 = 10 449 374 683  
S4 = 12 239 272 958  
S5 = 14 388 837 943  
S6 = 16 034 294 995  
S7 = 17 934 352 665  
S8 = 19 124 084 606  
S9 = 20 363 051 828  
S10 = 20 727 857 478  
S11 = 21 056 728 558

And we can check these values against Sylain Pion global result for N=21 :  
 $314\ 666\ 222\ 712 = S11 + 2*(S1 + S2 + S3 + S4 + S5 + S6 + S7 + S8 + S9 + S10)$   
Can we submit this extension using all our names / emails in your collection?  
Best Regards