Departamento de Matemática y Ciencia de la Computación

Licenciatura en Ciencia de la Computación

Laboratorio N°1

Combinaciones

Matemática Computacional
Profesor: Nicolas Thériault

Matías Daza A.
María Maldonado P.

Índice general

1. Introducción

En general si tenemos n objetos distintos, el número de combinaciones de tamaño r de estos objetos, que se denota por $C(n, r)$, donde 0 ≤ r ≤ n y se aplica la siguiente fórmula:

$$C(n, r) = \frac{n!}{r!(n-r)!}$$

2. Problema 1

Calcular los tres valores n!, r! y (n-r)! directamente desde la factorial:

$$k! = \prod_{i=1}^{k} i = 1 \cdot 2 \cdot 3 \cdot (k-1) \cdot k$$

y juntarlos en la expresión C(n,r) (cálculo exacto).

2.1. Algoritmo

Input : n,r
Output: C(n,r) = Combinatoria

```
1  multi ← 1
2  combinaciones ← 1
3  resta ← 1
4  aux_n ← 1
5  aux_r ← 1
6  i ← 1
7  if n < r then
8  |  return Error
9  END IF
10 for i to n do
11 |  aux_n ← aux_n · i
12 END FOR
13 for i to r do
14 |  aux_r ← aux_r · i
15 END FOR
16 for i to (n − r) do
17 |  resta ← resta · i
18 END FOR
19 multi ← resta · aux_r
20 combinaciones ← aux_n ÷ multi
21 return combinaciones
```

2.2. Descripción de la solución

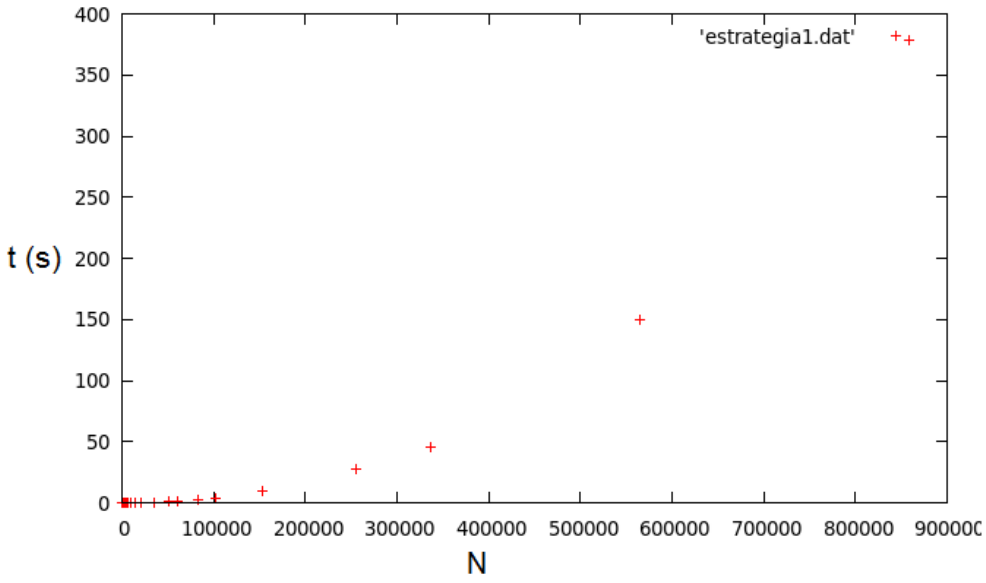
En este problema, se aplicó directamente la fórmula de la factorial, después el calculo de n! y r! se realiza por separado, luego se realiza la resta entre los términos y se finaliza al calcular directamente con la fórmula la cantidad de combinaciones.

2.3. Formulación del experimento

Experimento con n y r dadas y tiempo de ejecución.

n	r	Tiempo de ejecución (s)
5	2	0.000372
8	5	0.000276
20	13	0.000284
42	11	0.000316
151	109	0.000347
442	65	0.000286
1690	101	0.002534
3985	1875	0.119400
4234	503	0.009536
6744	222	0.027274
9031	7854	0.031615
13999	1209	0.082970
20145	253	0.175973
35000	111	0.522748
50999	333	1.099054
60785	23110	1.233500
83424	34343	2.281081
102444	345	4.478116
152542	10	10.501504
256000	12232	28.298876
336520	45553	46.735449
564545	6888	150.649641
858432	26	379.665983

2.4. Curvas de desempeño



2.5. Anális de la estrategia

De acuerdo a los resultados de la curva de desempeño, se puede observar que a medida que va creciendo n más tiempo se demora el algoritmo en calcular la solución. Esto es debido a que no se aplica ninguna estrategia matemática, sólo se calculan de manera directa las expresiones y luego se aplica la fórmula.

3. Problema 2

Utilizar las propiedades de la factorial y de la binomial (cancelaciones, simetrías) para simplificar el cálculo.

3.1. Algoritmo

Input : (n,r
Output: C(n,r) = Combinatoria

```
1  i ← 1
2  sucesor_n ← r + 2
3  fac_resta ← 1
4  fact_simp ← 1
5  total ← 1
6  if n < r then
7  |  return Error
8  END IF
9  for i to n − 1 do
10 |  fact_simp ← fact_simp · sucesor_n
11 |  sucesor_n ← sucesor_n + 1;
12 END FOR
13 for i to n do
14 |  fact_resta ← fact_resta · i
15 END FOR
16 total ← fact_simp ÷ fact_resta
17 return total
```

3.2. Descripción de la solución

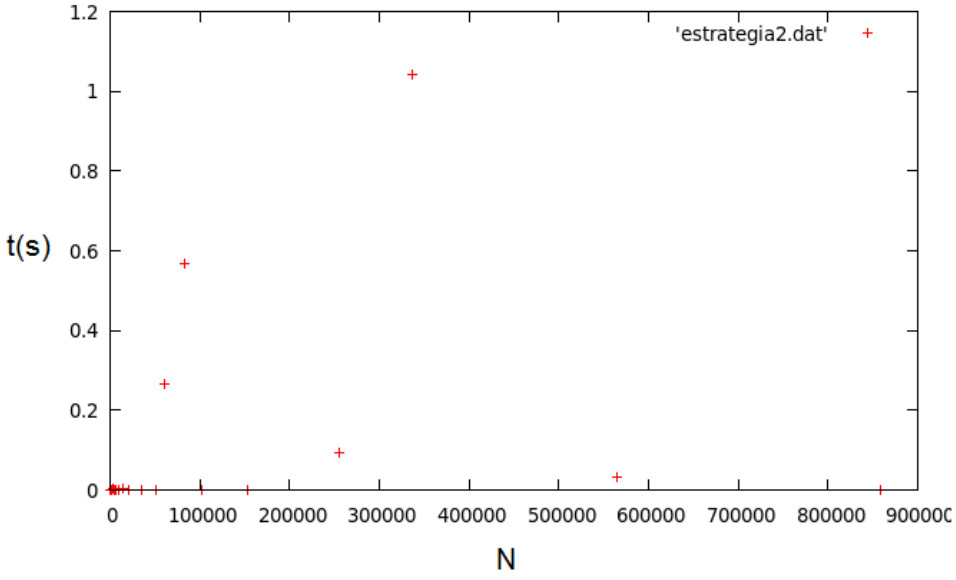
Para llevar a cabo las cancelaciones, el algoritmo utiliza los valores ingresados de n y r para utilizar la regla de simplifiación de la división entre factoriales, reduciendo así a la variable r. Luego r parte desde el número que simplificó hacia delante, logrando iterar las multiplicaciones hasta llegar a n y finalmente el factorial simplificado con el factorial de la resta entre (n - r), obteniendo la combinatoria apropiada.

3.3. Formulación del experimento

Experimento con n y r dadas y tiempo de ejecución.

n	r	Tiempo de ejecución (s)
5	2	0.000277
8	5	0.000277
20	13	0.000387
42	11	0.000276
151	109	0.000329
442	65	0.000348
1690	101	0.000454
3985	1875	0.004421
4234	503	0.000362
6744	222	0.000227
9031	7854	0.001840
13999	1209	0.002516
20145	253	0.000539
35000	111	0.000369
50999	333	0.000647
60785	23110	0.265432
83424	34343	0.566528
102444	345	0.000702
152542	10	0.000360
256000	12232	0.094321
336520	45553	1.041421
564545	6888	0.033435
858432	26	0.000309

3.4. Curvas de desempeño



3.5. Anális de la estrategia

De acuerdo a los resultados de la curva de desempeño, se puede observar que utilizando los mismos valores de n el tiempo de cálculo se reduce considerablemente. Esto es debido a que en esta estrategia se busca simplificar y cancelar los términos antes de realizar el cálculo, también mientras más cercanos sean los números mayor cantidad de cancelaciones se realizarán en comparación a dos números muy distantes.

4. Problema 3

Los cálculos exactos pueden dar números demasiado grandes para su computador. Una solución consiste en utilizar punto flotante para el cálculo de k! (en la primera estrategia).

4.1. Algoritmo

Se utiliza el algoritmo descrito en el problema 1.

4.2. Descripción de la solución

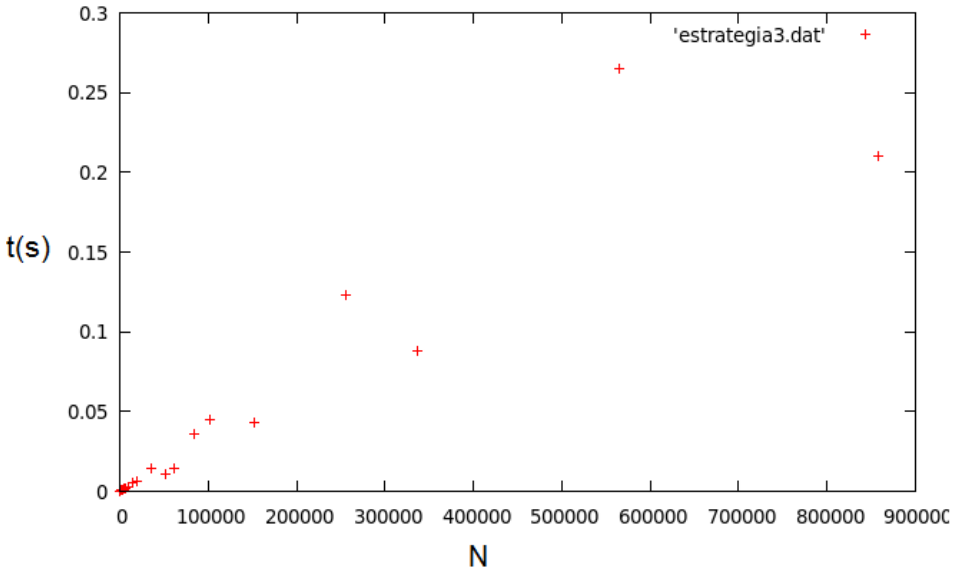
Se reemplazan las variables de tipo entero por varibles gmp de tipo flotante para que el cálculo sea más exacto.

4.3. Formulación del experimento

Experimento con n y r dadas y tiempo de ejecución.

n	r	Tiempo de ejecución (s)
5	2	0.000328
8	5	0.000310
20	13	0.000301
42	11	0.000389
151	109	0.000325
442	65	0.000468
1690	101	0.000758
3985	1875	0.000939
4234	503	0.001653
6744	222	0.002287
9031	7854	0.003182
13999	1209	0.005239
20145	253	0.006203
35000	111	0.014676
50999	333	0.011062
60785	23110	0.014550
83424	34343	0.035548
102444	345	0.044791
152542	10	0.042902
256000	12232	0.123040
336520	45553	0.088047
564545	6888	0.264705
858432	26	0.210112

4.4. Curvas de desempeño



4.5. Anális de la estrategia

De acuerdo a los resultados de la curva de desempeño, se puede observar que utilizando los mismos valores de n pero ahora en punto flotante, aplicados a la estrategia 1 (cálculo directo) el procesador puede dar cálculos más exactos para números grandes reduciendo el tiempo de cálculo considerablemente.

5. Problema 4

Combinar la aproximación numérica con la segunda estrategia. Por ejemplo:
$C(40,15) = \frac{40}{15} \cdot \frac{39}{14} \cdot \frac{38}{13} \cdots \frac{26}{1} \approx 2,667 \cdot 2,786 \cdot \cdot 2,923 \cdot \cdot 26,000 \approx 4,0225345x10^{10}$

5.1. Algoritmo

Se utiliza el algoritmo descrito en el problema 2.

5.2. Descripción de la solución

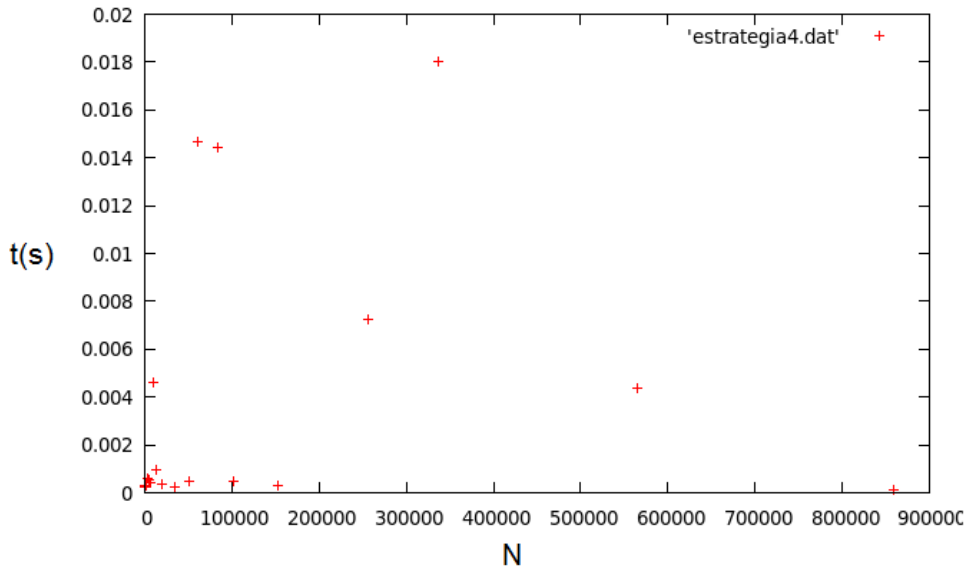
Ya que es una aproximación a la estrategia del problema 2, el resultado se puede expresar como la división iterativa entre n y r.

5.3. Formulación del experimento

Experimento con n y r dadas y tiempo de ejecución.

n	r	Tiempo de ejecución (s)
5	2	0.000261
8	5	0.000264
20	13	0.000269
42	11	0.000309
151	109	0.000322
442	65	0.000306
1690	101	0.000332
3985	1875	0.000589
4234	503	0.000553
6744	222	0.000405
9031	7854	0.004591
13999	1209	0.000948
20145	253	0.000372
35000	111	0.000233
50999	333	0.000468
60785	23110	0.014674
83424	34343	0.014409
102444	345	0.000481
152542	10	0.000312
256000	12232	0.007278
336520	45553	0.018045
564545	6888	0.004374
858432	26	0.000148

5.4. Curvas de desempeño



5.5. Anális de la estrategia

De acuerdo a los resultados de la curva de desempeño, se puede observar que combinando la estrategia de simplificación y cancelación pero con resultados en los reales, el cálculo de la combinatoria se realiza en un tiempo menor.

6. Problema 5

Utilizar una aproximación de la factorial, vía la fórmula de Stirling:

$$n! \approx \sqrt{2\pi}\frac{n^{n+\frac{1}{2}}}{\epsilon^n}$$

6.1. Algoritmo

Input : n,r
Output: C(n,r) = resultado

1 $i \leftarrow 1$
2 $rest \leftarrow (n - r)$
3 $aux\_n \leftarrow 1$
4 $aux\_r \leftarrow 1$
5 $aux\_rest \leftarrow 1$
6 $mult \leftarrow 1$
7 $mult\_2 \leftarrow 1$
8 $mult\_3 \leftarrow 1$
9 $resultado \leftarrow 1$
10 $e \leftarrow 2{,}71828182$
11 $\pi \leftarrow 3{,}14159265$
12 $mult\_sq\_n \leftarrow \sqrt{2 \cdot \pi}$
13 $mult\_sq\_r \leftarrow \sqrt{2 \cdot \pi}$
14 $mult\_sq\_rest \leftarrow \sqrt{2 \cdot \pi}$
15 $aux\_subs\_1 \leftarrow n \div e$
16 $aux\_subs\_2 \leftarrow r \div e$
17 $aux\_subs\_3 \leftarrow rest \div e$
18 **for** $i \leftarrow 1$ *to* n *do* **do**
19 | $aux\_n = aux\_n \cdot aux\_subs\_1$
20 END FOR
21 $mult = mult\_sq\_n \cdot aux\_n$
22 **for** $i \leftarrow 1$ *to* r *do* **do**
23 | $aux\_r = aux\_r \cdot aux\_subs\_2$
24 END FOR
25 $mult\_2 = mult\_sq\_r \cdot aux\_r$
26 **for** $i \leftarrow 1$ *to* $rest$ *do* **do**
27 | $aux\_rest = aux\_rest \cdot aux\_subs\_3$
28 END FOR
29 $mult\_3 = mult\_sq\_rest \cdot aux\_rest$
30 $mult\_3 = mult\_3 \cdot mult\_2$
31 $resultado = mult \cdot mult\_3$
32 *return* $resultado$

6.2. Descripción de la solución

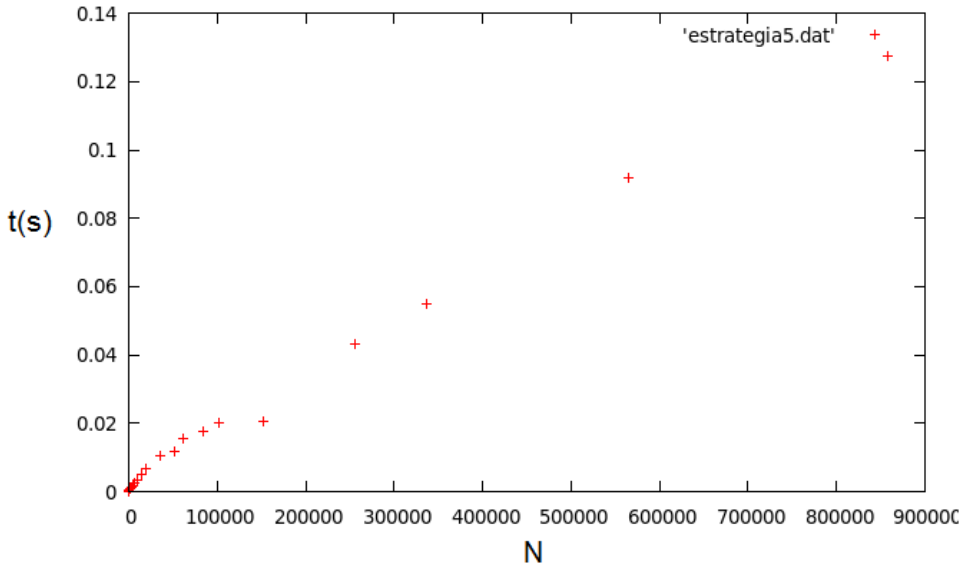
El algoritmo calcula $\sqrt{2 \cdot \pi \cdot n}$, $\sqrt{2 \cdot \pi}$, $\sqrt{2 \cdot \pi}$, luego se calcula $(\frac{n}{e})^n$, $(\frac{r}{e})^r$, $(\frac{rest}{e})^{rest}$ en la variable resultado se guarda el valor de $\frac{(\sqrt{2\cdot\pi}(\frac{n}{e})^n)}{(\sqrt{2\cdot\pi}(\frac{r}{e})^r)\cdot(\sqrt{2\cdot\pi}(\frac{rest}{e})^{rest})}$.

6.3. Formulación del experimento

Experimento con n y r dadas y tiempo de ejecución.

n	r	Tiempo de ejecución (s)
5	2	0.000185
8	5	0.000184
20	13	0.000183
42	11	0.000199
151	109	0.000232
442	65	0.000326
1690	101	0.000731
3985	1875	0.001452
4234	503	0.001596
6744	222	0.002438
9031	7854	0.003439
13999	1209	0.004929
20145	253	0.006833
35000	111	0.010354
50999	333	0.011961
60785	23110	0.015740
83424	34343	0.017525
102444	345	0.020002
152542	10	0.020617
256000	12232	0.043121
336520	45553	0.054982
564545	6888	0.091770
858432	26	0.127344

6.4. Curvas de desempeño



6.5. Anális de la estrategia

De acuerdo a los resultados de la curva de desempeño, se puede observar que en esta estrategia el cálculo iterativo de la fórmula de Stirling entrega valore más aproximados en la combinatoria de n y r, esta característica es muy importante para el cálculo de combinatoria de números grandes, sin embargo, el cómputo directo de ϵ y ϕ implican un mayor uso del procesador para el cálculo.

7. Conclusiones

El cálculo de una combinatoria puede parecer computacionalmente costosa, sin embargo, dependiendo de la estrategia utilizada para implementar los algoritmos ésta puede influir en un menor uso de memoria y un tiempo de ejecución considerablemente menor.

Por otro lado, como una combinatoria requiere de muchas multiplicaciones seguidas, la utilización de la librería GMP y variables de tipo float ayudan a que los cálculos se realicen de manera más eficiente.

8. Estructuras de Datos Utilizadas

Se utilizaron variables GMP las cuales trabajan con punteros.

9. Lectura de programa desde la consola

La lectura del programa se indica a continuación:

Para los 5 programas $p_i = 1, 2, 3, 4, 5$.

Para compilar:

- gcc -o lab1_p_i lab1_p_i.c -lgmp

Para ejecutar:

- ./lab1_p_i n r

10. Implementación

El algoritmo está implementado en lenguaje C.

11. Plataforma Computacional

El programa fue ejecutado en un computador con las siguientes características:

- **Procesador:** Intel® Core™ i3-350M CPU @ 2.27GHz 4
- **Memoria RAM:** 5,6 GiB
- **Tipo de SO:** 64 bits
- **Sistema Operativo:** Linux - Ubuntu 14.04 LTS

11.1. Librerías

- gmp.h
- stdio.h
- time.h

12. Bibliografía

- Librería GMP (http://gmplib.org).