Introduction:

[« suivant l'affirmation de Wikipédia sur cette conjecture de Goldbach :
« Le théorème des nombres premiers affirme qu'un entier ***m*** sélectionné aléatoirement d'une manière brute possède (**1/ Ln m**) chance d'être premier. Ainsi, si ***n*** est un grand entier pair et ***m***, un nombre compris entre **3** et ***n*/2**, alors on peut s'attendre à ce que la probabilité que ***m*** et ***n* – *m*** soient tous deux premiers soit égale à **1 / (Ln n . Ln m)** . Cet argument heuristique n'est pas rigoureux pour de nombreuses raisons; par exemple, on suppose que les événements que ***m*** et ***n* – *m*** **soient premiers sont** statistiquement indépendants **l'un de l'autre.** »

l'algorithme de Goldbach permet de prouver ***que cette affirmation est « Fausse »*** *pour une limite **n** fixée*, ainsi que la fonction asymptotique du TNP qu'il faudra modifier dans ces ***8 Fam (i)*** de nombres premiers **P' > 5**, représentant 26,666… % des entiers naturels non nuls; soit ***n / 3,75*** **.**

Car en effet, si on considère comme événement le calcul du nombre de nombres premier **q** ∈ **[*m ; n]*** ***il dépend statistiquement des congruences des entiers m*** ≤ ***n/2*** *limite fixée, autrement dit **q à pour antécédent m*** et donc si et seulement si **m** est un nombre premier **P'** ≤ ***n/2*** et tel que **m ≢ n [P]** alors **q a pour antécédent m = P',** il s'agit donc d'un seul et même événement dépendant de la **congruence de m ⇒ n – m** un nombre premiers **q.**
q ne peut être un multiple de **P,** mais le complémentaire de **m** par rapport à **n ;** c'est *une indépendance impossible.*

On peut déjà affirmer : que ces fonctions ci-dessus modifiées deviennent : (**1/**Ln **m** * Ln **n**) d'être premier et d'être ***non congru à n (mod P)*** , on obtient : **m / (Ln m . Ln n)** *le nombres d'entiers **m ≢ n [P]** qui précède un entier **m' = p'*** ⇔ **p' + q = n ,** car le nombre d'entiers **m** ***non congru à n mod P*** *est équivalent à* **m / Ln n,** *lorsque **m** → + ∞ ce qui implique les nombres premiers* **q** ∈ **[*m ; n]* ,** en utilisant l'algorithme de Goldbach **.**

Le raisonnement suivant est donc : les entiers **m ≢ n [P]** implique les **m = P' ou pas,** de ce fait, on ne peut pas supposer qu'il s'agisse de deux événements indépendants, car leur estimation est équivalente à **m / ln n,** caractérisé par le crible de Goldbach conséquence directe du TNP, qui re crible les nombres **m = P' ≤ n/2** criblés par Ératosthène selon le même principe.
Ce que l'on verra ci-après avec les deux cribles qui caractérisent les fonctions du TNP ainsi que les définitions relatives à ces algorithmes dans ces 8 Fam(i).**»]**

Pour en déduire en fonction d'une limite **n** fixée la fonction asymptotique **n / (Ln n . Ln 2n)** qui estime le nombre d'entiers naturels non nul **A' = m : non congruent modulo P et premier P' ⇒ q = 2n – A'.** *Peut on prouver que cette fonction pour toute limite **n ≥ 3 fixée ne peut être nulle, ie : il existe toujours P'+ q = 2n** ?*

<u>Document complémentaire sur l'illustration de l'algorithme de Goldbach et ses propriétés :</u>

*les entiers **A** sont les entiers non nuls en progression arithmétique de raison 30, dans les deux cribles, représentés par des 1 ou des 0. On va utiliser la propriété des congruences, sur les entiers **A**, congrus ou pas à **2n modulo P** : si **A** est congru **= 0** ; sinon **=1**. l'ensemble de ces **A ⇒ les P'≤ n.***

On veut prouver** que quelque soit une limite **N ≥ 150** vérifiée, tel que **2N** est somme de deux nombres premiers **P' + q ,** alors la conjecture serra prouvée pour la limite suivante **2N + 2 ,** par l'impossibilité d'infirmer la conjecture pour cette limite suivante **2N + 2 ;** à l'aide d'une famille d'entiers **A**, en progression arithmétique de raison **30** et de premier terme **(i)** ∈ **(1, 7, 11, 13, 17, 19, 23 ,29).
***La vérification de la conjecture pour une limite N⇒ N+1; «** plus généralement pour **N** ≥ **10^9,** La vérification de la conjecture pour une limite **N⇒ N+1+2+3+..n;** sur plusieurs limites successives et entraîne un effet boule de neige* **. »**

Le Décalage d'un rang des congruences sur les ,1, ou les ,0, ***relatif à la propriété de l'algorithme, vérifiera la conjecture pour 30(k+1) + 2i soit N= 15k + 1 = 3001 , 2N = 6002; Ce qui rend impossible l'infirmation de la conjecture:***
Donnez N = 15k = 3000 : on fixe la famille Fam(i) les 1sont les P' non congruent à P
fam1 crible G : et : Fam 1 Ératosthène EGCrible pour N=3000 : conjecture vérifiée pour tout n ≤ 3000.

[0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1] 19 ∓ 1 couples P'+ q = 6000.

On suppose que pour 2n + 2 = 6002 il n'y a pas de solution : Or, il suffit de regarder la limite N précédente qui a été vérifiée : 15(k-1) + 1 = 2985 + 1, avec sa Fam (i = 1) pour affirmer que la conjecture serra vérifiée pour la limite suivante 15k + 1 , tel que 30k + 2 est somme de deux premiers P'+ q !

Le décalage d'un rang des congruences sur leur successeur A+30, contredit la supposition!

ECrible les 1 sont premiers P' ≢ 2n[P] les 0 ≢ 2n[P] sont les multiple de P; les 1 et les 0 sont ≡ 2n [P]

[1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0]

Gcrible les 1 sont les entiers A ≢ 2n[P]

[0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0]

Ce qui donnera 17 couples P'+ q au minimum qui vérifieront 30(k+1) + 2i = 6002. Vérification avec EGCrible:

[0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1] réel 18 couples p+q = 2N. On peut montrer suivant cette propriété de l'algorithme de Goldbach, qu'il en sera de même sur les 14 limites successives suivantes. Ce ne sont que les antécédents décalés d'un rang ! *L.g* §

Pour 2n +2+2 = 6004 , il suffit de regarder la limite N précédente 15(k-1) + 2 = 2985 + 2, avec sa Fam (i = 17 ou 11 ou 23) qui a aussi été vérifiée.

N = 2987; fam 11 ; Ecrible P' = 1 ou 0 ⇔ ≢ 2n [P] et en dessous Gcrible Entier A = 1 ⇔ ≢ 2n [P] et P≤ √2n

avec le décalage des congruences des A non congrus ***1*** *ou* ***0 se reportent sur les*** *A+30 = P'=* ***1****ou* ***1*** *congrus ou pas* ***(30(k-1)+ 2) [P].***

E

[1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0]

G

[0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1]

vérification EG:

[0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0] réel 21 couples p+q = 2N

---

*nous venons de vérifier le décalage récurrent des congruences des A=**1** ou A=**0 non congruent (mod P) qui se reportent** sur leur successeur **A'+30** =* ***1****ou* ***1*** ***, qui ont pour antécédent A,*** *le contraire contredirait le TNP et Le TFA, (cqfd)*

*La conjecture se montre avec l'une des 8 Fam(i) en fonction de la forme de **N** et pour toute limite **N ≥ 150** :*

*1) les nombres A **≢ 2n[P]** ⇔ **q** ∈ [N;2N] est donc défini pour **2N + 2** à une exception près !*
*Car les restes R de **30(k-1)+2i** par P ne peuvent plus être utilisés du simple fait qu'ils ne correspondent pas **aux nouveaux** R de **30k + 2i** par **P** qui eux seront les mêmes à une exception près lorsque **N** augmente de 1.*
*Si on utilisait les R de **30(k -1)+ 2i et les** R de 30k + 2i, le cardinal du nombre de nombres premiers **q** serait faux pour la limite **2N+2** qui a déjà été fixé et vérifié avec les nombres **P** qui ont criblés la limite **N – 15** précédente.*

2) Si ce décalage d'un rang des congruences sur leur successeur A'+30 ne s'effectuait pas, le TFA serait contredit ! Du fait que:
*L'égalité : **(2n – A = B)≠ ((2n+30) – (A+30) = B)** ce qui est idiot, **B** est toujours le même; si **B** est un nombres premiers **P'** il pourrait être multiple de **P** si ce décalage des non congruences ne se décalaient pas sur lui , et inversement un multiples de **P** pourrait se trouver premiers si la congruence ne se décalait pas sur lui, pour la limite suivante **15(k+1)+i**.*
*Car : Suivant l'exemple ci dessus avec la **fam 11**, nous avons le troisième élément **71 ≢ 2N[P]** pour **2N = (2*2987)** et tel que **5974 – 71 = q = 5903** premier et **101 est congruent** a P pour cette limite N, il est évident que pour la limite sui-*

vante, N =15(k+1)+2 = 3002 ⇒ ***2N = 6004*** *–* ***(71+30)*** *=* ***5903 = q*** *il est bien toujours le même, par conséquent et suivant cette propriété de décalage d'un rang,* ***101*** *serra par obligation* ***non congruent*** *à P, le contraire est impossible !*

Conclusion:

a) L'égalité récurrente implique le décalage d'un rang des congruences tel que :
(2n – A = B)⇔ ((2n+30) – (A+30) = B).

b) Si un entier ***A*** *en progression arithmétique de raison 30, de premier terme* ***i ,*** *premiers P' ou pas, tel que A ≢* ***2N[P] précède A+30 = P' congrus ou pas à 2N [P] , suite à ce décalage d'un rang des congruences P' vérifiera la conjecture pour la limite N=15k, 15(k+1)+ i*** *et ainsi de-suite pour plusieurs limites successives.*

Autrement dit*, cela implique lors des limites précédentes* ***N – 15k*** *au rang* ***n – k****, on connaît déjà les couples qui vérifieront la conjecture pour cette limite* ***N*** *. Ce qui rend impossible l'infirmation de la conjecture pour la limite suivante* ***N+1*** *sans avoir besoin de la cribler,* ***2N + 2*** *serra donc somme de deux nombres premiers. Ce que l'on peut montrer ci-dessous en* ***d).***

c) L'impossibilité de réutiliser les restes ***R*** *de* ***2n*** *par* ***P*** *pour la limite suivante =* ***15(k+1)+ 1*** *avec sa* ***fam(i), l'égalité récurrente et le décalage d'un rang des congruences*** *confirme la véracité de la conjecture de Goldbach. On en déduit alors, qu'il est impossible de l'infirmer, «en moyenne générale, pratiquement jusqu'au double des limites successives.»*

Vérifions le troisième cas :

pour 15(k-1)+ 3 = 2988 ; soit pour 2n +2+2+2 = 6006 , on regarde la limite précédent n = 15(k-1)+ 3 = 2985 + 3 et sa fam (i = 7, ou 29 ou 17 ou 19 ou 13 ou 23):

N = 2988; fam 7 ; Ecrible P' = 1 ou 0 ⇔ ≢ 2n [P] et en dessous Gcrible Entier A = 1 ⇔ ≢ 2n [P] et P ≤ √2n

E:{7,37,67,97,127,157…...2977} = N // 30 donne le nombres d'entiers (mod 30) < 2988 qui sont criblés, soit 99.

[1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0]

G:

[0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0]

vérification EG , N = 3003:

[0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0]

d) On pourrait supposer: qu'en serait il de la limite **N = 15k + 6 = 3006 ?**

Étant donné la conjecture vérifiée pour tout **N ≤ 3000** il suffit de s'apercevoir qu'au rang 6; c'est à dire la fam **(i)** qui a vérifiée N – 90 =15(k-6)+6 = **2916** avec l'une des 6 fam **(i)** = (**1,11 ; 13,29 ; 19,23**):
il y avait obligatoirement des entiers **A** ≢ ***2N[P] qui précédaient P' au rang 6; car cela est dû à la propriété récurrente de l'algorithme, afin de préserver l'égalité du point*** *a) ; Le contraire implique que* **N ≤ 3000 n'a pas de solution.**

Existe t'il suite au décalage des congruences, une limite, où il est impossible de vérifier qu'il existera toujours cette limite de longueur ***n*** *contenant une densité de* ***A≢ 2N[P]*** *précédents* ***A' = P'****, ie : que la fonction* $\dfrac{n}{(\ln(n)*\ln(2n))}$ *est donc toujours positive, ou encore avec* $\dfrac{\pi(n)}{\ln(2n)}$ *; car on prend en compte les entiers* ***A ≢ 2n[p],*** *contenant des nombres premiers* ***p'*** *pour une même limite* ***n ≥ 3*** *fixée. On peut par conséquent pour* ***n > 149*** *estimer un minimum de couples* ***P'+q*** *, en ne prenant en compte que* ***π(n)*** *est* ***G(n),*** *puisque la fonction* ***G(n)*** *ne tient compte que des entiers* ***A, premiers ou pas ≤ n et non congrus à 2n[P].***

La réponse est non*, du simple fait que pour toute limite* ***N*** *vérifiée le décalage des congruences repousse la véracité de la conjecture jusqu'au double de cette limite* ***N ; en moyenne générale la fonction*** *du nombre de* ***A≢ 2N[P] implique le nombre de rangs successifs des limites N suivantes et par fam (i), qui seront vérifiées.***

vérification:

E: *pour N =* ***2916*** ***et fam 1*** *{1,31,61,91,121,151,181,211 ..etc;* ***A*** *précède au rang 6 un* ***P'= 1 ou 1*** ***donc :*** *tout entier* ***A= 1 ou 0 non congruent à P, qui précède un nombre premier p' ou p' congru ou pas au rang 6, vérifiera la conjecture pour la limite N= 15(k+6) + 6 = 3006.***

[1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0] ***remarque importante****, ce sont principalement des* ***0 multiples de p*** *qui vont permettre de vérifier la conjecture pour* ***N=15(k+6)+6 ; ce qui garantie une certaine densité de couples p'+ q = 6012*** *et permet d'affirmer la fonction d'estimation de ce nombre de couples.*

G:

[0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0] *soit au minimum 23 sur 29 couples* ***P'****+q, qui vérifieront* ***2N = 6012*** ***; il n'y a rien d'aléatoire.***

Donnez N: 3006

[1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1]. Nombres P' non congru 2n[pi], de 1 à 3006 famille 1 : 29 ----- 0.01

En règle générale la conjecture est vérifiée sur plusieurs limites 15(k+1,2,...n) successives lorsque ***n*** *tend vers l'infini,* ***avec un effet boule de neige****, ce qui rend son infirmation impossible,* ***donc elle est vraie****.*
La fonction d'estimation du nombre de couples ***p+q = 2n*** *est une conséquence directe du TNP caractérisée par les deux fonctions relatives à ces deux algorithmes, elle sera par conséquent toujours positive :*

G(n) nombre d'entiers A $\not\equiv$ 2n[P] $\leq$ n, vaut **~** $\lim_{n\to+\infty}$ ¿ $\frac{n}{(\log 2n)}$

Le TNP dit que $\pi(N) = \frac{N}{(\log N)} + o\left(\frac{N}{\log N}\right)$, donc le nombre de nombres premiers dans $]N,2N]$ vaut

$$\pi(2n) - \pi(n) = \left(\frac{2n}{\log(2N)} - \frac{n}{\log N}\right) + o\left(\frac{n}{\log n}\right)$$

$$= N \times \frac{2}{\log(2N)} - \frac{1}{\log N} + o\left(\frac{n}{\log n}\right)$$

$$= N \times \frac{2\log n - \log(2n)}{\log(2n)\log n} + o\left(\frac{n}{\log n}\right)$$

$$= \frac{n}{\log 2n} + o\left(\frac{n}{\log n}\right)$$

Conséquence des deux fonctions du TNP*, on en déduit que le nombre d'entiers* ***A$\not\equiv$ 2n [P]*** *qui précèdent un entier* ***A'=P'$\Leftrightarrow$P'+ q = 2n*** *est équivalent à:*

$$\frac{n}{(\ln(n) * \ln(2n))} \quad \textit{lorsque} \quad \lim_{n\to+\infty} ¿$$

Autre estimation heuristique : Le nombre de couples de nombres premiers (p', q) ou le nombre de nombres premiers A = P' **non congrus à 2n modulo P** de 1 à n ; vaut environ, lorsque la $\lim_{n\to+\infty}$ ¿ pour n =15k + n' , avec n' $\in$ [0 ; 14] et **k** entier naturel non nul :

simplement $\frac{\pi(n)}{\ln \pi(n)}$ mais plus précisément : $C_2 \frac{G(n)}{\ln G(n)}$ **;** où G(n) est la fonction qui compte le nombre

d'entiers A de 1 à n , non congruent modulo **P** et $C_2 \approx 1,320323$ constante des premiers jumeaux.

****

Annexe 1

En fonction de la limite **N = 15k** fixée, on fixe lune des **8 Fam (i)** correspondante à la forme de **N**

Pour **N** de la forme **15k**, on peut choisir n'importe la quelle des 8 Fam.

Le programme est fixé avec une limite **N** début = 3000000000 et Fin = 3000000330 il progressera de raison 15,
Il n'y a que la fam(i) à rentrer à la demande:

Pour toute Limite **N = 15k + n'**, avec **n'** ∈ **[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14]** on rentre la Fam(i) correspondante

N = 15k+1; Fam =(1,13,19); **2n = 30k + 2**
N = 15k+2; Fam =(11,17,23); **2n = 30k + 4**
N = 15k+3; Fam =(7,29,13,23,17,19); **2n = 30k + 6**
N = 15k+4; Fam =(1,7,19); 2n = **30k + 8**
N = 15k+5; Fam =(1,7,13,19)**;** **2n = 30k + 10**
N = 15k+6; Fam =(1,11,13,19,23,29); **2n = 30k + 12**
N = 15k+7; Fam =(1,7,13); 2n = **30k + 14**
N = 15k+8; Fam =(17,23,29); 2n = **30k + 16**
N = 15k+ 9; Fam =(1,7,11,17,19,29); 2n = **30k + 18**
N = 15k+10; Fam =(11,29,17,23); 2n = **30k + 20**
N = 15k+11; Fam =(11,23,29); **2n = 30k + 22**
N = 15k+ 12; Fam =(1,7,11,13,17,23); 2n = **30k + 24**
N = 15k+ 13; Fam =(7,13,19); 2n = **30k + 26**
N = 15k+ 14; Fam =(11,17,29); 2n = **30k + 28**

N = 15(k+1) + 0; ***l'une des 8 Fam*** ***2n = 30 (k + 1)***

****

Connaissant le nombre de premiers P' de 1 à N :

on peut aussi faire une estimation du nombre de couples ***(p+q) par fam(i)*** *comme ci dessous :*

En utilisant les deux fonctions ***de π(n)= n /log n*** *et* ***G(n)= n /log 2n ;*** *puis on fait la différence.*

Extrait pour n = 900000000153 et fam 29 : (π(n) – G(n)) * log 15

Vaut ~ (4247753690 – 4030718929) * log15 = 587 741 028,17 .

valeur réelle pour n = 900000000003 pour les 6 fam(i) = 3 506 586 451

valeur réelle pour n+15 = 900000000018 pour les 6 fam(i) = 3 859 788 463

suivant la propriété récurrente du décalage des congruences sur leur successeur, permettant d'affirmer que la conjecture sera vérifiée jusqu'au double de la limite précédente et l'effet boule de neige :

la valeur réelle pour n + 3 = 1800000000003 pour les 6 fam(i) = 6 232 499 956 soit une augmentation de plus de80 %.

Avec la fonction 3 du TNP : $\dfrac{n}{(\ln(n)*\ln(2n))}$

*On obtient pour les 6 fam(i) : **1800000000003 / (log 1800000000003 * log 3600000000006) = 2 206 258 445,..** donc bien en dessous de la valeur réelle du nombres de couples qui décomposent **1800000000306** en somme de deux premiers. Il faut donc affiner une fonction correctrice , ou encore vérifier avec la fonction Li(n).*

```
/home/gilbert/Programmes/E_G_Crible_8.fam
famille : 7 limite : 900000000003
Nombre premiers criblés famille 7 plus petits que 900000000003: 4247765037 time 88.9337
Nombre couples p+q=2N criblés famille 7 : 584451924 time 88.3397
famille : 7 limite : 900000000018
Nombre premiers criblés famille 7 plus petits que 900000000018: 4247765037 time 88.4114
Nombre couples p+q=2N criblés famille 7 : 643313462 time 88.3579
famille : 13 limite : 900000000003
Nombre premiers criblés famille 13 plus petits que 900000000003: 4247747130 time 88.3892
Nombre couples p+q=2N criblés famille 13 : 584419162 time 88.3658
famille : 13 limite : 900000000018
Nombre premiers criblés famille 13 plus petits que 900000000018: 4247747130 time 88.418
Nombre couples p+q=2N criblés famille 13 : 643269721 time 88.3887
famille : 17 limite : 900000000003
Nombre premiers criblés famille 17 plus petits que 900000000003: 4247750241 time 88.2892
Nombre couples p+q=2N criblés famille 17 : 584429560 time 88.4109
famille : 17 limite : 900000000018
Nombre premiers criblés famille 17 plus petits que 900000000018: 4247750241 time 88.2832
Nombre couples p+q=2N criblés famille 17 : 643269602 time 88.4004
famille : 19 limite : 900000000003
Nombre premiers criblés famille 19 plus petits que 900000000003: 4247736977 time 88.4251
Nombre couples p+q=2N criblés famille 19 : 584431628 time 88.2779
famille : 19 limite : 900000000018
Nombre premiers criblés famille 19 plus petits que 900000000018: 4247736977 time 88.4319
Nombre couples p+q=2N criblés famille 19 : 643274016 time 88.2858
famille : 23 limite : 900000000003
Nombre premiers criblés famille 23 plus petits que 900000000003: 4247760141 time 88.4074
Nombre couples p+q=2N criblés famille 23 : 584447749 time 88.405
famille : 23 limite : 900000000018
Nombre premiers criblés famille 23 plus petits que 900000000018: 4247760141 time 88.4398
Nombre couples p+q=2N criblés famille 23 : 643278470 time 88.4233
famille : 29 limite : 900000000003
Nombre premiers criblés famille 29 plus petits que 900000000003: 4247753690 time 88.3619
Nombre couples p+q=2N criblés famille 29 : 584406428 time 88.4679
famille : 29 limite : 900000000018
Nombre premiers criblés famille 29 plus petits que 900000000018: 4247753690 time 88.3736
Nombre couples p+q=2N criblés famille 29 : 643283642 time 88.3661

Process returned 0 (0x0)   execution time : 2135.713 s
Press ENTER to continue.
```

Pour N = 1 800 000 000 003 et les 6 fam (i) :N = 15k +8 ;

```
/home/gilbert/Programmes/E_G_Crible_8.fam
famille : 7 limite : 1800000000003
Nombre premiers criblés famille 7 plus petits que 1800000000003: 8278483964 time 180.082
Nombre couples p+q=2N criblés famille 7 : 1038741440 time 179.662
famille : 13 limite : 1800000000003
Nombre premiers criblés famille 13 plus petits que 1800000000003: 8278464010 time 179.654
Nombre couples p+q=2N criblés famille 13 : 1038729111 time 179.616
famille : 17 limite : 1800000000003
Nombre premiers criblés famille 17 plus petits que 1800000000003: 8278507604 time 179.341
Nombre couples p+q=2N criblés famille 17 : 1038725736 time 179.57
famille : 19 limite : 1800000000003
Nombre premiers criblés famille 19 plus petits que 1800000000003: 8278470115 time 179.635
Nombre couples p+q=2N criblés famille 19 : 1038768470 time 179.779
famille : 23 limite : 1800000000003
Nombre premiers criblés famille 23 plus petits que 1800000000003: 8278491448 time 179.743
Nombre couples p+q=2N criblés famille 23 : 1038780381 time 179.67
famille : 29 limite : 1800000000003
Nombre premiers criblés famille 29 plus petits que 1800000000003: 8278478813 time 179.837
Nombre couples p+q=2N criblés famille 29 : 1038754368 time 181.409

Process returned 0 (0x0)   execution time : 2171.941 s
Press ENTER to continue.
```

Fam(i) = 17 ; 23 ou 29 ; fami (i) complémentaire 29 ;23 ; ou 17

/home/gilbert/Programmes/E_G_Crible_8.fam

```
famille : 17 limite : 6300000000008
Nombre couple criblés famille 17 entre 6300000000008 et 12600000000016: 39580985
92 time 824.844
famille : 17 limite : 6300000000023
Nombre couple criblés famille 17 entre 6300000000023 et 12600000000046: 3265477006 time 825.73
famille : 17 limite : 6300000000038
Nombre couple criblés famille 17 entre 6300000000038 et 12600000000076: 3265490379 time 5122.51
famille : 17 limite : 6300000000053
Nombre couple criblés famille 17 entre 6300000000053 et 12600000000106: 3269483317 time 5118.28
famille : 17 limite : 6300000000068
Nombre couple criblés famille 17 entre 6300000000068 et 12600000000136: 3457567758 time 5122.74
famille : 17 limite : 6300000000083
Nombre couple criblés famille 17 entre 6300000000083 et 12600000000166: 3483496099 time 9416.15
famille : 17 limite : 6300000000098
Nombre couple criblés famille 17 entre 6300000000098 et 12600000000196: 3919471963 time 9415.66
famille : 17 limite : 6300000000113
Nombre couple criblés famille 17 entre 6300000000113 et 12600000000226: 3267557009 time 9416.19
famille : 17 limite : 6300000000128
Nombre couple criblés famille 17 entre 6300000000128 et 12600000000256: 3312706978 time 13709.2
famille : 17 limite : 6300000000143
Nombre couple criblés famille 17 entre 6300000000143 et 12600000000286: 3329585639 time 13707.3
famille : 17 limite : 6300000000158
Nombre couple criblés famille 17 entre 6300000000158 et 12600000000316: 3265473910 time 13708.8
famille : 23 limite : 6300000000008
Nombre couple criblés famille 23 entre 6300000000008 et 12600000000016: 3958164527 time 18009.5
famille : 23 limite : 6300000000023
Nombre couple criblés famille 23 entre 6300000000023 et 12600000000046: 3265519244 time 18008.8
famille : 23 limite : 6300000000038
Nombre couple criblés famille 23 entre 6300000000038 et 12600000000076: 3265493217 time 18005.7
famille : 23 limite : 6300000000053
Nombre couple criblés famille 23 entre 6300000000053 et 12600000000106: 3269454523 time 22300.9
famille : 23 limite : 6300000000068
Nombre couple criblés famille 23 entre 6300000000068 et 12600000000136: 3457503462 time 22302.1
famille : 23 limite : 6300000000083
Nombre couple criblés famille 23 entre 6300000000083 et 12600000000166: 3483554742 time 22301.5
famille : 23 limite : 6300000000098
Nombre couple criblés famille 23 entre 6300000000098 et 12600000000196: 3919368191 time 26599.2
famille : 23 limite : 6300000000113
Nombre couple criblés famille 23 entre 6300000000113 et 12600000000226: 3267590611 time 26597.8
famille : 23 limite : 6300000000128
Nombre couple criblés famille 23 entre 6300000000128 et 12600000000256: 3312790281 time 26597.2
famille : 23 limite : 6300000000143
Nombre couple criblés famille 23 entre 6300000000143 et 12600000000286: 3329594866 time 30890.6
famille : 23 limite : 6300000000158
Nombre couple criblés famille 23 entre 6300000000158 et 12600000000316: 3265404337 time 30892.1

Process returned 0 (0x0)   execution time : 41510.160 s
Press ENTER to continue.
```

On augmente la limite N de 15. On utilise pour cela, les trois fam (i) = 11 ; 17 et 23
nombre de premiers criblée ⇒ nombre de couples p+q = 2 * 4500 000 000 002 à 2 * 4500 000 000 107

/home/gilbert/Programmes/E_G_Crible_8.fam

```
famille : 11 limite : 4500000000002
Nombre premiers criblés famille 11 plus petits que 4500000000002: 20020233398 time 517.285
Nombre couples p+q=2N criblés famille 11 : 2607250497 time 534.803
famille : 11 limite : 4500000000017
Nombre premiers criblés famille 11 plus petits que 4500000000017: 20020233398 time 519.102
Nombre couples p+q=2N criblés famille 11 : 2414789785 time 538.586
famille : 11 limite : 4500000000032
Nombre premiers criblés famille 11 plus petits que 4500000000032: 20020233398 time 517.226
Nombre couples p+q=2N criblés famille 11 : 2387299132 time 4831.89
famille : 11 limite : 4500000000047
Nombre premiers criblés famille 11 plus petits que 4500000000047: 20020233398 time 4812.35
Nombre couples p+q=2N criblés famille 11 : 2387419091 time 4831.23
famille : 11 limite : 4500000000062
Nombre premiers criblés famille 11 plus petits que 4500000000062: 20020233398 time 4812.37
Nombre couples p+q=2N criblés famille 11 : 2881665432 time 4833.6
famille : 11 limite : 4500000000077
Nombre premiers criblés famille 11 plus petits que 4500000000077: 20020233398 time 4812.28
Nombre couples p+q=2N criblés famille 11 : 2392874730 time 4831.57
famille : 11 limite : 4500000000092
Nombre premiers criblés famille 11 plus petits que 4500000000092: 20020233398 time 4812.37
Nombre couples p+q=2N criblés famille 11 : 2543413744 time 9127.56
famille : 11 limite : 4500000000107
Nombre premiers criblés famille 11 plus petits que 4500000000107: 20020233398 time 9108.35
Nombre couples p+q=2N criblés famille 11 : 2390888872 time 9125.38
famille : 17 limite : 4500000000002
Nombre premiers criblés famille 17 plus petits que 4500000000002: 20020230303 time 9106.88
Nombre couples p+q=2N criblés famille 17 : 2607237962 time 9125.06
famille : 17 limite : 4500000000017
Nombre premiers criblés famille 17 plus petits que 4500000000017: 20020230303 time 9103.24
Nombre couples p+q=2N criblés famille 17 : 2414764165 time 9122.85
famille : 17 limite : 4500000000032
Nombre premiers criblés famille 17 plus petits que 4500000000032: 20020230303 time 9107.6
Nombre couples p+q=2N criblés famille 17 : 2387361906 time 13418.7
famille : 17 limite : 4500000000047
Nombre premiers criblés famille 17 plus petits que 4500000000047: 20020230303 time 13402.3
Nombre couples p+q=2N criblés famille 17 : 2387391058 time 13418.2
famille : 17 limite : 4500000000062
Nombre premiers criblés famille 17 plus petits que 4500000000062: 20020230303 time 13402
Nombre couples p+q=2N criblés famille 17 : 2881684253 time 13418.8
famille : 17 limite : 4500000000077
Nombre premiers criblés famille 17 plus petits que 4500000000077: 20020230303 time 13403.6
Nombre couples p+q=2N criblés famille 17 : 2392893286 time 13417.8
famille : 17 limite : 4500000000092
Nombre premiers criblés famille 17 plus petits que 4500000000092: 20020230303 time 13403.8
Nombre couples p+q=2N criblés famille 17 : 2543414474 time 17711.5
famille : 17 limite : 4500000000107
Nombre premiers criblés famille 17 plus petits que 4500000000107: 20020230303 time 17694
Nombre couples p+q=2N criblés famille 17 : 2390985893 time 17713.4
famille : 23 limite : 4500000000002
Nombre premiers criblés famille 23 plus petits que 4500000000002: 20020235931 time 17700.1
Nombre couples p+q=2N criblés famille 23 : 2607229116 time 17718.3
famille : 23 limite : 4500000000017
Nombre premiers criblés famille 23 plus petits que 4500000000017: 20020235931 time 17694.9
Nombre couples p+q=2N criblés famille 23 : 2414889851 time 17712.8
famille : 23 limite : 4500000000032
Nombre premiers criblés famille 23 plus petits que 4500000000032: 20020235931 time 17699.9
Nombre couples p+q=2N criblés famille 23 : 2387362595 time 22010.5
famille : 23 limite : 4500000000047
Nombre premiers criblés famille 23 plus petits que 4500000000047: 20020235931 time 21992.9
Nombre couples p+q=2N criblés famille 23 : 2387344606 time 22011.9
famille : 23 limite : 4500000000062
Nombre premiers criblés famille 23 plus petits que 4500000000062: 20020235931 time 21990
Nombre couples p+q=2N criblés famille 23 : 2881730868 time 22011.8
famille : 23 limite : 4500000000077
Nombre premiers criblés famille 23 plus petits que 4500000000077: 20020235931 time 21991.4
Nombre couples p+q=2N criblés famille 23 : 2392862314 time 22013.1
famille : 23 limite : 4500000000092
Nombre premiers criblés famille 23 plus petits que 4500000000092: 20020235931 time 21993.5
Nombre couples p+q=2N criblés famille 23 : 2543470424 time 26306.2
famille : 23 limite : 4500000000107
Nombre premiers criblés famille 23 plus petits que 4500000000107: 20020235931 time 26287.3
Nombre couples p+q=2N criblés famille 23 : 2390956946 time 26305.6

Process returned 0 (0x0)   execution time : 28635.434 s
Press ENTER to continue.
```

Entier A≢ 2n[P] ≤ 4500 000 000 001 familles (1,13,19)

```
/home/gilbert/Programmes/goldbach N+15.8fam
famille : 1 limite : 4500000000001
Nombre premiers criblés famille 1 entre 4500000000001 et 9000000000002: 19054830203 time 516.748
famille : 1 limite : 4500000000016
Nombre premiers criblés famille 1 entre 4500000000016 et 9000000000032: 19054830203 time 516.284
famille : 13 limite : 4500000000001
Nombre premiers criblés famille 13 entre 4500000000001 et 9000000000002: 19054778517 time 521.255
famille : 13 limite : 4500000000016
Nombre premiers criblés famille 13 entre 4500000000016 et 9000000000032: 19054778517 time 518.809
famille : 19 limite : 4500000000001
Nombre premiers criblés famille 19 entre 4500000000001 et 9000000000002: 19054827970 time 518.593
famille : 19 limite : 4500000000016
Nombre premiers criblés famille 19 entre 4500000000016 et 9000000000032: 19054827970 time 4813.52

Process returned 0 (0x0)   execution time : 3595.292 s
Press ENTER to continue.
```

Entier A≢ 2n[P] ≤ 4500 000 000 001 familles (11,23,17)

```
/home/gilbert/Programmes/goldbach N+15.8fam
famille : 11 limite : 4500000000002
Nombre premiers criblés famille 11 entre 4500000000002 et 9000000000004: 19054822236 time 518.961
famille : 11 limite : 4500000000017
Nombre premiers criblés famille 11 entre 4500000000017 et 9000000000034: 19054822236 time 518.898
famille : 17 limite : 4500000000002
Nombre premiers criblés famille 17 entre 4500000000002 et 9000000000004: 19054819131 time 528.022
famille : 17 limite : 4500000000017
Nombre premiers criblés famille 17 entre 4500000000017 et 9000000000034: 19054819131 time 513.511
famille : 23 limite : 4500000000002
Nombre premiers criblés famille 23 entre 4500000000002 et 9000000000004: 19054843692 time 519.252
famille : 23 limite : 4500000000017
Nombre premiers criblés famille 23 entre 4500000000017 et 9000000000034: 19054843692 time 4816.17

Process returned 0 (0x0)   execution time : 3637.440 s
Press ENTER to continue.
```

Entier A≢ 2n[P] ≤ 4500 000 000 00, 8 familles (1,7,11,13,17,19,23,29)

/home/gilbert/Programmes/goldbach N+15.8fam

```
famille : 1 limite : 4500000000000
Nombre premiers criblés famille 1 entre 4500000000000 et 9000000000000: 19054859771 time 520.599
famille : 1 limite : 4500000000015
Nombre premiers criblés famille 1 entre 4500000000015 et 9000000000030: 19054859771 time 517.325
famille : 1 limite : 4500000000030
Nombre premiers criblés famille 1 entre 4500000000030 et 9000000000060: 19054859772 time 521.145
famille : 7 limite : 4500000000000
Nombre premiers criblés famille 7 entre 4500000000000 et 9000000000000: 19054822236 time 520.376
famille : 7 limite : 4500000000015
Nombre premiers criblés famille 7 entre 4500000000015 et 9000000000030: 19054822236 time 519.442
famille : 7 limite : 4500000000030
Nombre premiers criblés famille 7 entre 4500000000030 et 9000000000060: 19054822236 time 4817.07
famille : 11 limite : 4500000000000
Nombre premiers criblés famille 11 entre 4500000000000 et 9000000000000: 19054778517 time 4817.44
famille : 11 limite : 4500000000015
Nombre premiers criblés famille 11 entre 4500000000015 et 9000000000030: 19054778517 time 4818.9
famille : 11 limite : 4500000000030
Nombre premiers criblés famille 11 entre 4500000000030 et 9000000000060: 19054778517 time 4818.06
famille : 13 limite : 4500000000000
Nombre premiers criblés famille 13 entre 4500000000000 et 9000000000000: 19054819131 time 4816.02
famille : 13 limite : 4500000000015
Nombre premiers criblés famille 13 entre 4500000000015 et 9000000000030: 19054819131 time 4813.18
famille : 13 limite : 4500000000030
Nombre premiers criblés famille 13 entre 4500000000030 et 9000000000060: 19054819131 time 4812.78
famille : 17 limite : 4500000000000
Nombre premiers criblés famille 17 entre 4500000000000 et 9000000000000: 19054827970 time 4815.79
famille : 17 limite : 4500000000015
Nombre premiers criblés famille 17 entre 4500000000015 et 9000000000030: 19054827970 time 9111.93
famille : 17 limite : 4500000000030
Nombre premiers criblés famille 17 entre 4500000000030 et 9000000000060: 19054827970 time 9113.38
famille : 19 limite : 4500000000000
Nombre premiers criblés famille 19 entre 4500000000000 et 9000000000000: 19054843692 time 9108.96
famille : 19 limite : 4500000000015
Nombre premiers criblés famille 19 entre 4500000000015 et 9000000000030: 19054843692 time 9116.63
famille : 19 limite : 4500000000030
Nombre premiers criblés famille 19 entre 4500000000030 et 9000000000060: 19054843692 time 9108.93
famille : 23 limite : 4500000000000
Nombre premiers criblés famille 23 entre 4500000000000 et 9000000000000: 19054825261 time 9109.13
famille : 23 limite : 4500000000015
Nombre premiers criblés famille 23 entre 4500000000015 et 9000000000030: 19054825260 time 9109.94
famille : 23 limite : 4500000000030
Nombre premiers criblés famille 23 entre 4500000000030 et 9000000000060: 19054825260 time 9109.24
famille : 29 limite : 4500000000000
Nombre premiers criblés famille 29 entre 4500000000000 et 9000000000000: 19054830203 time 13403.8
famille : 29 limite : 4500000000015
Nombre premiers criblés famille 29 entre 4500000000015 et 9000000000030: 19054830203 time 13404.6
famille : 29 limite : 4500000000030
Nombre premiers criblés famille 29 entre 4500000000030 et 9000000000060: 19054830203 time 13403.4

Process returned 0 (0x0)   execution time : 14512.536 s
Press ENTER to continue.
```

Pour limite **N** = 4500 000 000 000 ; puis N /30 = 150 000 000 000, nombre d’entiers A **par Famille** (i) ∈(1,7,11,13,17,19,23,29)

Famille(i) = 13
P’ = 20 020 216 296 **< N/30 = 150 000 000 000**
A’ ≢ 2n [P] = 19 054 859 771 **< N/30**
Rapport du nombre de A par le nombre de **P’** ; par famille :
150 000 000 000 / 20 020 216 296 = **7,4924**

Rapport du nombre de A par le nombre de **A’** ≢ 2n [P] par famille : = **7,872**

Rapport du nombre de **P’** par les **P’**≢ 2n[P], pour **famille (13)** variation du rapport: de **8,386 à 6,8987** lorsque **N** augmente de raison 15
Rapport du nombre de **A’** ≢ 2n [P] par les **P’**≢ 2n[P], par famille variation : de **7,9816 à 6,5661** lorsque N augmente de raison 15

Estimation Heuristique : ln (4500 / 2) = **7,7186855** ce qui devrait donner une estimation de couples **P’+ q** par rapport au nombre de **P’ ou par** rapport au nombre de **A’ ≢ 2n [P]** qui précèdent les (**A’+30) = P’** pour la limite précédente n-15 donnant le nombre de solutions pour la limite suivante limite 2n , 2n+2 ...; *« ainsi que cet effet boule de neige »*
(« *il faut analyser la courbe de cette fonction asymptotique par tranche de 1000 mds* »)

<u>Annexe 1) :</u>

Programme des algorithmes {Ératosthène / Goldbarch} : Pour changer les fam(i) et la valeur de début et fin, on modifie les lignes de codes 156 et 157; puis on désactive les familles de 160 à 167 avec les deux //.

```
// -*- compile-command: "/usr/bin/g++ -g goldbachs.cc" -*-
#include <vector>
#include <iostream>
#include <cmath>
#include <stdlib.h>
#include <time.h>
using namespace std;
// fill Erathosthene sieve crible for searching primes up to 2*crible.size()*32+1
// crible is a (packed) bit array, crible[i] is true if 2*i+1 is a prime
// crible must be set to true at startup
void fill_crible(vector<unsigned> &crible, unsigned p)
{
  crible.resize((p - 1) / 64 + 1);
  unsigned cs = crible.size();
  unsigned lastnum = 64 * cs;
  unsigned lastsieve = int(std::sqrt(double(lastnum)));
  unsigned primesieved = 1;
  crible[0] = 0xfffffffe; // 1 is not prime and not sieved (2 is not sieved)
  for (unsigned i = 1; i < cs; ++i)
    crible[i] = 0xffffffff;
  for (; primesieved <= lastsieve; primesieved += 2)
  {
    // find next prime
    unsigned pos = primesieved / 2;
    for (; pos < cs; pos++)
    {
      if (crible[pos / 32] & (1 << (pos % 32)))
        break;
    }
```

```
    // set mutiples of (2*pos+1) to false
    primesieved = 2 * pos + 1;
    unsigned n = 3 * primesieved;
    for (; n < lastnum; n += 2 * primesieved)
    {
      pos = (n - 1) / 2;
      crible[(pos / 32)] &= ~(1 << (pos % 32));
    }
  }
}
unsigned nextprime(vector<unsigned> &crible, unsigned p)
{
  // assumes crible has been filled
  ++p;
  if (p % 2 == 0)
    ++p;
  unsigned pos = (p - 1) / 2, cs = crible.size() * 32;
  if (2 * cs + 1 <= p)
    return -1;
  for (; pos < cs; ++pos)
  {
    if (crible[pos / 32] & (1 << (pos % 32)))
    {
      pos = 2 * pos + 1;
      // if (pos!=nextprime(int(p)).val) CERR << "error " << p << endl;
      return pos;
    }
  }
  return -1;
}

typedef unsigned long long ulonglong;

size_t ECrible(const vector<ulonglong> &premiers, ulonglong n, int fam, vector<bool> &crible, size_t lencrible)
{ //on va contruire un tableau de 1 modulo 30 et rappeler les premiers p
  int cl = clock();
  // size_t lencrible = n / 30,
  size_t nbpremiers = premiers.size(); //on va contruire un tableau de 1 modulo 30 en divisant N par 30
  //vector<bool> crible(lencrible, true);                    // on rappelle les nombres premiers p d'Eratotene ci dessus
  // ulonglong n2=2*n;
  vector<ulonglong> indices(nbpremiers);
  for (size_t i = 0; i < nbpremiers; ++i)
  {
    ulonglong p = premiers[i];
    ulonglong produit;
    int GM[] = {7, 11, 13, 17, 19, 23, 29, 31}; // on va calculer le produit de p par un element du groupe GM
    for (size_t j = 0; j < sizeof(GM) / sizeof(int); j++)
    {
      produit = p * GM[j]; // calcul du produit, jusqu'a ce que le produit soit égale à fam modulo 30
      if (produit % 30 == fam)
      {
        produit /= 30; // puis on va va calculer l'indice, afin de commencer à cribler de l'indice à n/30 et on réitère
        break;
      }
    }
    indices[i] = produit;
  }
  ulonglong nslices = lencrible / 1500000, currentslice = 0;
  if (nslices == 0)
    nslices = 1;
```

```
  for (; currentslice < nslices; ++currentslice)
  {
    size_t slicelimit = currentslice + 1;
    slicelimit = slicelimit == nslices ? lencrible : (currentslice + 1) * (lencrible / nslices);
    for (size_t i = 0; i < nbpremiers; ++i)
    {
      ulonglong p = premiers[i];
      size_t index;
      for (index = indices[i]; index < slicelimit; index += p)
        crible[index] = 0;
      indices[i] = index;
    }
  }
  size_t total = 0;
  for (size_t index = 0; index < lencrible; ++index)
    total += int(crible[index]);
  cout << "Nombre premiers criblés famille " << fam << " plus petits que " << n << ": " << total << " time " << (clock()
- cl) * 1e-6 << endl;
  return total; // à la fin du crible on return le résultat est le temps mis
}

size_t GCrible(const vector<ulonglong> &premiers, ulonglong n, int fam, vector<bool> &crible, size_t lencrible)
{
  int cl = clock();
  //size_t lencrible = n / 30,
  size_t nbpremiers = premiers.size(); //on va contruire un tableau de 1 modulo 30 en divisant N par 30
  //vector<bool> crible(lencrible, true);                    // on rappelle les nombres premiers p d'Eratotene ci dessus
  ulonglong n2 = 2 * n;
  vector<ulonglong> indices(nbpremiers);
  for (size_t i = 0; i < nbpremiers; ++i)
  {
    ulonglong p = premiers[i];
    ulonglong reste = n2 % p; // on calcule le reste de 2n par p
    if (reste % 2 == 0)
      reste += p;
    ulonglong pi2 = 2 * p;
    while (reste % 30 != fam) // tant que le reste += p n'est pas = à Fam % 30 on rajoute 2*p
      reste += pi2;
    reste /= 30; // on ensuite on va calculer l'indice pour commencer à cribler le tableau de 1.1.1.... avec p, de l'indice à
n/30
    indices[i] = reste;
  }
  ulonglong nslices = lencrible / 1500000, currentslice = 0;
  if (nslices == 0)
    nslices = 1;
  for (; currentslice < nslices; ++currentslice)
  {
    size_t slicelimit = currentslice + 1;
    slicelimit = slicelimit == nslices ? lencrible : (currentslice + 1) * (lencrible / nslices);
    for (size_t i = 0; i < nbpremiers; ++i)
    {
      ulonglong p = premiers[i];
      size_t index;
      for (index = indices[i]; index < slicelimit; index += p)
        crible[index] = 0;
      indices[i] = index;
    }
  }
  size_t total = 0;
  for (size_t index = 0; index < lencrible; ++index)
```

```
    total += int(crible[index]); // le criblage du tableau de 1 modulo 30 jusqu'a n/30 (1.1.1.1...etc) est fini on va retourner
le résultat
  cout << "Nombre couples p+q=2N criblés famille " << fam << " : " <<total << " time " << (clock() - cl) * 1e-6 <<
endl;
  return total;
}

int main(int argc, char **argv)
{
  vector<unsigned> temp;
  ulonglong debut = 3000;
  ulonglong fin = 3225;

  vector<int> familles;
  familles.push_back(1);
  familles.push_back(7);
  familles.push_back(11);
  familles.push_back(13);
  familles.push_back(17);
  familles.push_back(19);
  familles.push_back(23);
  familles.push_back(29);

  for (int i = 0; i < familles.size(); i++)
  {
    int fam = familles[i];

    for (ulonglong limite = debut; limite < fin; limite += 15)
    {
      cout << "famille : " << fam << " limite : " << limite << endl;
      double sqrt2N = unsigned(std::sqrt(2 * double(limite)));
      fill_crible(temp, sqrt2N);
      vector<ulonglong> premiers;
      for (ulonglong p = 7; p <= sqrt2N;)
      {
        premiers.push_back(p);
        p = nextprime(temp, p);
        if (p == unsigned(-1))
          break;
      }

      size_t lencrible = limite / 30;
      vector<bool> crible(lencrible, true);
      ECrible(premiers, limite, fam, crible, lencrible);
      GCrible(premiers, limite, fam, crible, lencrible);
    }
  }
}
```