STIRLING-Formel für approximative Berechnung von n!

Ac 2022

Nicht immer sind natürliche Zahlen zur Berechnung der Fakultät n! effizient geeignet.
Selbst der JAVA-Datentyp BigInteger tut sich bei sehr großen Zahlen mitunter schwer .
Z.B. dauert die Berechnung von 1 000 000 ! auf derzeit üblichen PCs mehrere Minuten !!
(Allerdings gibt es auch Optimierungen von Peter Luschny, welche für diese Aufgabe lediglich Sekunden beanspruchen).

Als Ausweg können **Approximationen** mithilfe der sog. **Stirling-Formel** , benannt nach James Stirling (1692–1770), dienen:

<u>Stirling-Reihenentwicklung für n! :</u>

$$\boxed{\ln(n!) = n\cdot(\ln(n)-1) + \frac{\ln(2\pi n)}{2} + \frac{1}{12n} - \frac{1}{360n^3} + \frac{1}{1260n^5} - \frac{1}{1680n^7} + \ldots + \frac{B_{2k}}{(2k-1)\cdot 2k\cdot n^{2k-1}} + \ldots}$$

B_{2k} ist hierbei die so genannte "**2k-te Bernoullizahl**":

Genaue Definition der $\mathbf{B_k}$ (rekursiv) :

$$\boxed{B_0 = 1; \quad B_k = \frac{-1}{k+1}\cdot\sum_{j=0}^{k-1}\binom{k+1}{j}\cdot B_j \quad \text{für } k \geq 1}$$

<u>Achtung</u> : Für **ungerade** Indizes k $\geq$ **3** gilt: $\mathbf{B_k = 0}$

Einige Bernoullizahlen :
$B_0 = 1$ $B_1 = -1/2$ $B_2 = 1/6$ $B_4 = -1/30$ $B_6 = 1/42$ $B_8 = -1/30$ $B_{10} = 5/66$ $B_{12} = -691/2730$
$B_{14} = 7/6$ $B_{16} = -3617/510$ $B_{18} = 43867/798$ $B_{20} = -174611/330$ $B_{22} = 854513/138$
$B_{24} = -236364091/2730$ $B_{26} = 8553103/6$ $B_{28} = -23749461029/870$ $B_{30} = 8615841276005/14322$

Je genauer die Approximation mit der Stirling-Reihenentwicklung sein soll, umso mehr Reihenglieder (und damit B_{2k} - Terme) werden benötigt und müssen ermittelt werden!

Bricht man die STIRLING-Reihe nach dem zweiten Summanden ab, so ist der <u>absolute</u> Fehler $|f| < 1/(12n)$.

Für n > 1000 genügt der erste Summand, um den <u>relativen</u> Fehler kleiner als 1% zu halten !
Für sehr große n wird der relative Fehler noch kleiner.

<u>Hinweis:</u> Noch genauer sind die Kettenbruch-Approximationen von Stieltjes !

Die Berechnung des n! - Wertes bei sehr großen n-Werten wird im folgenden erläutert :

Wie erhält man n! , wenn n sehr groß ist ?

Zuerst muss eine Approximation für z = ln(n!) gemäß der Stirling-Reihe berechnet werden.
Wir beschränken uns dabei auf die Glieder der Reihe bis einschließlich 1/(12n) .

Beispiel: ln(1 000 000 !) ≈ 1000000 · (ln(1000000) - 1) + ln(2π·1000000)/2 + 1/(12·1000000)
≈ 12815518,384658

Der "normale Weg" wäre dann die Bildung von e^z, was n! liefern würde.

Leider aber ist es kaum möglich, e^ln(n!) für **sehr große n** in angemessener Zeit zu berechnen bzw. mit dem normalen Datentyp “double” darzustellen, weil diese Zahl für normale Rechner zu groß ist.
Für das Beispiel 1 000 000 ! ergäbe sich ca. 8,26393112·10^5565708 .
Der Datentyp “double” endet bei ca. 10^308 .

Man muss daher eine Umformung vornehmen, um von ln(n!) auf das gesuchte n! zu schließen .
Man rechnet daher zunächst um auf die Basis 10: **lg(n!) = ln(n!) / ln(10)**

Für obiges Beispiel ergibt sich: **lg(1000000!)** ≈ 12815518,384658 / ln(10) ≈ **5565708,91718664**

Jetzt hat man eine neue Dezimalzahl, deren Vorkommateil (Vk) den Exponenten der Zehnerpotenz von n! anzeigt. Der Exponent ist also expo := Vk(lg(n!)). Man notiert dann einfach den String 10^expo .
Für obiges Beispiel notiert man also **10^5565708** .

Die Mantisse steckt im Nachkommateil (Nk) von lg(n!), und zwar **berechnet** man :
mantisse = 10^Nk(lg(n!)) . Für obiges Beispiel notiert man 10^0,91718664 ≈ **8,2639296**.

Der gesuchte Gesamtstring setzt sich dann zusammen aus: “mantisse” und ”·10^expo” .
Das Ergebnis ist dann **1000000! ≈ 8,26393·10^5565708** .

Fazit: Approximationsformel für n! nach ausgerechnetem ln(n!) :

Erst $\boxed{\mathbf{lg}(\mathbf{n!}) = \mathbf{ln}(\mathbf{n!})/\mathbf{ln(10)}}$ berechnen, dann $\boxed{n! \approx 10^{\mathrm{Nk}(\lg(n!))} \cdot 10^{\mathrm{Vk}(\lg(n!))}}$

beim 1.Faktor muss die gesamte Potenz ausgerechnet werden, beim 2. nur der Exponent !

Beispiele (Anwendung der obigen Formeln) :

ln(56!) ≈ 56·(ln(56)-1) + ln(112π)/2 + 1/672 – 1/63221760 ≈ 172,3527971 (Stirling-Reihenentwicklung)
Also ist lg(56!) = 172,3527971/ln(10) ≈ 74,85186872 und wegen 10^0,85186872 ≈ 7,109986 gilt dann: **56! ≈ 7,109986·10^{74}**

ln(500!) ≈ 500·(ln(500)-1) + ln(1000π)/2 + 1/6000 – 1/4,5*10^{10} ≈ 2611,330458
Also ist **500! ≈ 1,2201368·10^{1134}**

ln(12000!) ≈ 12000·(ln(12000)-1) + ln(24000π)/2 + 1/6,2208*10^{14} – 1/3,1352832*10^{23} ≈ 100717,5584
Also ist **12000! ≈ 1,2018584·10^{43741}**

Ebenso
50000! ≈ 3,3473205·10^{213236}