Prime number Research Program (PrP)

door Theo Kortekaas

Beschrijving en gebruikers handleiding PrP versie 2.0

Doel van het programma is om priemgetallen in het getallendomein van 0 tot 2^{64} te genereren en te manipuleren. Het programma is bedoeld om te werken in een 32-bits Windows omgeving op een normale desktop of laptop computer met een hoofdgeheugen van minimaal één gigabyte.
Onder manipuleren kan worden verstaan: priemgetallen tellen; priemtweelingen opsporen; priemgaten tellen en statistieken van priemgaten vervaardigen; priem k-tuple clusters opsporen .
De methode die gebruikt wordt om priemgetallen te genereren is de zeef van Eratosthenes. Zie website:
nl.wikipedia.org/Zeef_van_Eratosthenes

De zeef van Eratosthenes

Om met de zeef van Eratosthenes priemgetallen tot 2^{64} te genereren zijn de priemgetallen tot 2^{32} nodig (wortel 2^{64} is 2^{32}). Verder is een getallenrij nodig van 0 tot 2^{64}. We zullen zien dat dit niet mogelijk is (met de huidig beschikbare computers) en welke mogelijkheden we dan wel kunnen benutten.

Als eerste actie stelt het PrP programma een tabel samen met alle priemgetallen tot 2^{32}. (Dat gebeurt ook met behulp van de zeef van Eratosthenes.) Dat zijn er 203.280.221. Deze priemgetallen kunnen tot 32 bits groot zijn; dus vier bytes per stuk. Een complete tabel zou ruim 800 megabytes beslaan. Omdat deze tabel frequent moet worden gebruikt moet deze continue in het computer-geheugen staan. Dat legt een te groot beslag op het computer geheugen.
Er is voor gekozen om niet de priemgetallen zelf, maar de priemgaten in een tabel op te nemen. Het grootste priemgat dat voorkomt bij priemgetallen tot 2^{32} is 320. Omdat alle priemgaten (behoudens één uitzondering) een even getal vormen kan de waarde van een priemgat, (bij het opbergen) gedeeld worden door twee. De grootste waarde die moet worden opgenomen is dan 320/2 = 160. Deze waarde kan ruimschoots in één byte worden geplaatst. Bij het regenereren van het priemgetal moet de opgeborgen waarde dan weer vermenigvuldigd worden met twee.

Een nadeel van deze methode is dat de priemgetallen alleen kunnen worden teruggevonden door de tabel van voor af aan te doorlopen. Dit is voor het

toepassen van deze tabel bij de zeef van Eratosthenes geen bezwaar.
Voor de eerste twee priemgetallen (twee en drie) geldt dat het priemgat geen even getal is. Daarom is er voor gekozen om deze twee priemgetallen geheel op te nemen, elk in een eigen byte. De index voor de elementen van deze tabel begint bij nul. Er is voor gekozen om de index gelijk te laten lopen met het volgnummer van het priemgetal. Daarom krijgt de de eerste byte de waarde 0 (index 0); de tweede byte bevat 2 (index 1 en 2 is het eerste priemgetal); de derde byte bevat 3 (index 2 en 3 is het tweede priemgetal).
Het eerste deel van de tabel ziet er nu zó uit:

0 Eerste byte, bewust nul, omdat de index van deze byte nul is.
2 (eerste priemgetal)
3 (tweede priemgetal)
1 (* 2 priemgat tussen 3 en 5)
1 (* 2 priemgat tussen 5 en 7)
2 (* 2 priemgat tussen 7 en 11)
1 (* 2 priemgat tussen 11 en 13)
2 (* 2 priemgat tussen 13 en 17) enz.
Dit zijn dus acht bytes voor de eerste zeven priemgetallen.

In totaal beslaat de tabel nu 203.280.222 bytes voor alle 203.280.221 priemgetallen kleiner dan $2^{32.}$ Deze tabel moet passen in het computergeheugen en een copy moet op schijf kunnen worden geplaatst.
Verder is voor de zeef nodig een “gesorteerde rij” van getallen; dus een rij getallen op volgorde van die getallen; deze rij moet in het computergeheugen staan en “afgestreept” kunnen worden; dus bij elk getal hoort een indicator die aan of uit gezet kan worden. De kleinst mogelijke indicator is een bit; echter een bit is niet zomaar afzonderlijk te adresseren; extra instructies zijn nodig om bits te manipuleren.
Uit oogpunt van ruimte in het computer geheugen is een bit de optimale keuze; echter uit oogpunt van snelheid kan beter gekozen worden voor een byte, die afzonderlijk te adresseren is en simpel aan en uit (op nul of een) gezet kan worden.
In de ideale situatie zou de gehele getallen-reeks van 0 tot 2^{64} in het computer-geheugen worden geplaatst. Echter met de beperkingen van de huidige desktop of laptop computers (draaiend onder 32-bitsWindows) kan niet meer dan een getallen-reeks van circa maximaal één miljard getallen worden opgenomen: één byte per getal zou dus maximaal één gigabyte vergen.
Dit betekent dat van het getallen-bereik van 0 tot 2^{64} slechts stukje bij beetje tegelijk kan worden behandeld: de zeef wordt *gesegmenteerd.*

Zoekruimte

Het programma biedt de mogelijkheid om een getallenreeks op te geven. Die reeks wordt aangeduid met "*het getallenbereik*" en bestaat uit de reeks getallen tussen een "van" nummer en een "tot" nummer. Dit *getallenbereik* vormt de *zoekruimte* waarin naar priemgetallen gezocht wordt; daarom wordt dit *getallenbereik* ook wel aangeduid met *zoekruimte*.
De *zoekruimte* kan bij nul beginnen, maar dat is niet perse nodig. De *zoekruimte* kan zo groot worden gedefinieerd als wenselijk is, maar naarmate de *zoekruimte* groter is, wordt ook de tijd die nodig is om deze ruimte te doorzoeken groter.

De gesegmenteerde zeef

De oorspronkelijke opzet van de zeef van Eratosthenes is dat de priemgetallen in het begin van de getallenrij worden gebruikt om veelvouden van die priemgetallen verderop in de getallenrij "weg te strepen" zodat uiteindelijk alleen priemgetallen over blijven in de gehele getallenrij. Dit werkt alleen als de getallenrij begint bij nul.
Omdat het PrP programma de beschikking heeft over een tabel met alle priemgetallen tot 2^{32} is er geen noodzaak meer om de getallenrij te laten beginnen met nul.
Indien de opgegeven *zoekruimte* te groot is voor het computer-geheugen dan wordt de *zoekruimte gesegmenteerd*. De grootte van een segment kan worden opgegeven.

Voor een deel van de getallenrij wordt in het computergeheugen ruimte gereserveerd ter grootte van een segment; een byte per getal. Er worden bewerkingen op uitgevoerd. Dan wordt voor een volgend deel van de getallenrij ruimte gereserveerd, enzovoorts. Tot de gehele *zoekruimte* is behandeld.
De resultaten van de bewerkingen worden gerapporteerd per segment en in totaal voor de gehele *zoekruimte*.

Gebruikers handleiding

Het Prime Research Program kan worden gedownload via www.tonjanee.home.xs4all.nl/downld.html. Nadat, na het downloaden, de file-extensie veranderd is van .pgm naar .exe kan het programma worden overgebracht naar een map, bij voorkeur een eigen map voor priemgetallen, hierna aangeduid met "*de map*". *De map* moet gevestigd zijn op een schijf waar nog minimaal circa 250 megabyte vrije ruimte is. Nadat het programma gestart is verschijnen de eerste berichten van het programma op het display.

Het display

Op het display worden twee display-gebieden getoond. Het bovenste deel wordt gebruikt om gegevens in te voeren. Het onderste deel is een zogenaamde "scroll-area". Hierop worden aanwijzingen en boodschappen van het programma getoond en de resultaten van de berekeningen weergegeven. Daarbij wordt elke nieuwe regel onderaan toegevoegd, terwijl alle regels een plaats naar boven opschuiven en een regel bovenaan verdwijnt als de regel buiten de "scroll-area" komt.

Online log en logbestand

De regels die op het display worden vertoond worden ook in een online log geplaatst. Deze log biedt plaats aan ongeveer 12.000 regels. Wanneer de log vol is worden nieuwe regels van voor af aan in de log geplaatst waarbij de oude informatie wordt overschreven. Dit wordt "wrap-around" genoemd. Indien dit plaatsvindt wordt een mededeling hierover opgenomen in de log. De gegevens van de online log worden ook in een logbestand op schijf geplaatst. Hiervoor moet voldoende ruimte aanwezig zijn in *de map*.

Na het starten van het programma is het mogelijk om de taal te wijzigen (voorlopig alleen Nederlands en Engels) en kan het wegschrijven in het logbestand op schijf worden onderdrukt.

Inlezen priemgetallenbestand

Als eerste actie moet nu het priemgetallenbestand met de naam "PGTABLE.TAB" worden ingelezen of opgebouwd. Dit wordt in gang gezet door "Instelllen" te selecteren en daarna "Bouwen of inlezen Priemgat tabel". Indien het priemgetallenbestand reeds aanwezig is in *de map* dan wordt dit

bestand ingelezen. Dit duurt enige seconden. Wanneer bij het inlezen een probleem optreedt, dan wordt een foutmelding gegeven en stopt het programma. In dat geval moet het priemgetallenbestand handmatig worden verwijderd. Daarna kan het programma weer worden gestart. Indien geen probleem optreedt wordt de boodschap “Priemgetallenbestand ingelezen” gegeven.
Indien het priemgetallenbestand niet aanwezig is in *de map* dan wordt dit bestand aangemaakt. Dit kan enige minuten duren. Tijdens het aanmaken wordt de voortgang gerapporteerd.

Specificeren getallenbereik

Nadat het priemgetallenbestand succesvol is aangemaakt of ingelezen, kan het *getallenbereik* worden gespecificeerd. Dit gebeurt door ”Instellen” te selecteren en daarna “Bepaal begin en eind van getallenbereik”. Er wordt een image van een numeriek toetsenbord getoond. Daarop kan met de muis een getal worden ingetoetst. Dit is het begin nummer van het getallenbereik. Het nummer mag minimaal nul zijn en maximaal 18.446.744.073.709.550.000. Het nummer moet in ieder geval even zijn. Als het nummer goed is ingetoetst dan kan de OK toets worden ingedrukt.
Hierna kan eindnummer op dezelfde wijze als het beginnummer worden ingetoetst. Dit nummer moet groter zijn dan het begin nummer en moet minimaal 16 zijn en mag maximaal 18.446.744.073.709.551.616 (dit is 2^{64}) zijn. Ook dit nummer moet een even getal zijn en wordt afgesloten met de OK toets.
Nu kan de grootte van een segment worden ingevoerd. Het nummer dat wordt ingetoetst geeft het aantal “mebibytes” aan dat als zeef-gebied wordt gebruikt (één “mebibyte” afkorting MiB is 1.024 x 1.024 bytes = 1.048.576 bytes; dit is conform de IEC-standaard die in 1998 is ingevoerd). De grootte van een segment is minimaal 1 MiB en maximaal 1024 MiB.
Opmerking:
Het verschil tussen het begin nummer en het eind nummer vormt de *zoekruimte;* de grootte van de *zoekruimte* is bepalend voor de doorlooptijd van de verschillende acties die kunnen worden uitgevoerd. Afhankelijk van computersnelheid, geheugengrootte en segmentgrootte kan het programma van 10 miljoen tot 60 miljoen getallen per seconde beoordelen op priem zijn. Bij een *zoekruimte* van bijv. een biljoen (10^{12}) kan een doorlooptijd verwacht worden van enkele uren tot meer dan een dag!

De Acties

Er zijn zeven verschillende acties die kunnen worden uitgevoerd op de *zoekruimte*. Deze acties vertegenwoordigen het echte werk.
Indien eenmaal een *zoekruimte* is ingesteld, kunnen een of meerdere acties in willekeurige volgorde achter elkaar worden uitgevoerd op dezelfde *zoekruimte*. Voor elke actie geldt dat resultaten per segment worden gerapporteerd en totalen per zoekruimte. De resultaten worden getoond op het display in het "scroll-gebied", waarbij elke nieuwe regel onderaan wordt toegevoegd, terwijl alle regels een plaats naar boven opschuiven en een regel bovenaan verdwijnt als de regel buiten het "scroll-gebied" komt. De regels die op het display worden vertoond worden ook in een online log geplaatst. Deze log biedt plaats aan ongeveer 12.000 regels. Wanneer de log vol is worden nieuwe regels van voor af aan in de log geplaatst waarbij de oude informatie wordt overschreven. Alle regels van de online log worden ook in het logbestand op schijf geplaatst. Er gaat dus geen informatie verloren.

Actie 1. Tellen priemgetallen en priemtweelingen

Alle priemgetallen in de *zoekruimte* worden geteld, evenals alle priemtweelingen en de aantallen worden per segment gerapporteerd. Een priemtweeling is een tweetal priemgetallen die twee verschillen; bijvoorbeeld 11 en 13 vormen een priemtweeling evenals 17 en 19. Een priemtweeling bestaat uit twee priemgetallen en wordt daarom als twee geteld in aantallen priemtweelingen. De aantallen priemgetallen bevatten ook de priemgetallen die behoren tot een priemtweeling.
De priemgetallen 3, 5 en 7 vormen een priem-drieling! Ze worden niet meegeteld in het aantal priemtweelingen, maar uiteraard wel in het aantal priemgetallen.
Opmerking:
Indien zich aan het begin van de zoekruimte een priemtweeling bevindt waarvan de kleinste niet binnen de zoekruimte valt, dan wordt deze niet meegeteld in aantallen priemtweelingen. Het zelfde geldt indien zich aan het einde van de zoekruimte een priemtweeling bevindt waarvan de grootste niet binnen de zoekruimte valt.
Actie 1 kent de snelste uitvoeringstijd. Bij een grote *zoekruimte* kan de uitvoeringstijd toch nog zeer lang zijn. Om bij langdurige tel-acties geen resultaten verloren te laten gaan bij (al dan niet geplande) onderbrekingen, worden per segment ook cumulatieve aantallen gerapporteerd.

Actie 2. Berekenen en tonen priemgetallen

Alle priemgetallen die binnen de zoekruimte vallen worden getoond in de scroll-area van de display; een regel per priemgetal. Dit gebeurt zo snel dat afzonderlijke priemgetallen nauwelijks zijn te onderscheiden. Deze actie kan alleen zinvol worden gebruikt indien de zoekruimte beperkt wordt en de resultaten worden gebruikt uit het logbestand.

Actie 3. Berekenen en tonen priemtweelingen

Alle priemtweelingen die binnen de *zoekruimte* vallen worden getoond in de scroll-area van de display; twee regels per priemtweeling met een separatie regel. Dit gebeurt zo snel dat afzonderlijke priemgetallen nauwelijks zijn te onderscheiden. Deze actie kan alleen zinvol worden gebruikt indien de zoekruimte beperkt wordt en de resultaten worden gebruikt uit het logbestand. Vanaf nul is de eerste priemtweeling die getoond wordt: 11 en 13. Priemgetallen 3, 5 en 7 worden niet tot een priemtweeling gerekend.
Opmerking:
Indien zich aan het begin van de zoekruimte een priemtweeling bevindt waarvan de kleinste niet binnen de zoekruimte valt, dan wordt deze niet getoond en niet meegeteld in aantallen priemtweelingen. Het zelfde geldt indien zich aan het einde van de zoekruimte een priemtweeling bevindt waarvan de grootste niet binnen de zoekruimte valt.

Actie 4. Berekenen en tonen priemgaten

De priemgaten tussen alle priemgetallen binnen de *zoekruimte* worden berekend en deze priemgaten worden per priemgatgrootte geteld. Per segment worden de aantallen getoond en op het einde van de *zoekruimte* worden de totalen per priemgatgrootte getoond. Bovendien wordt voor elke grootte het priemgetal vermeld, waarbij deze grootte voor het eerst werd aangetroffen. Dit is dan wel binnen de *zoekruimte*. Het priemgat aan het begin van de *zoekruimte* en het priemgat aan het einde van de *zoekruimte* worden niet meegeteld.

Actie 5. Bereken eenzaamste priemgetallen

Naarmate priemgetallen groter worden komen ze verder uit elkaar te liggen. Maar de priemgetallen liggen nogal willekeurig verspreid over alle getallen. Soms liggen ze dicht bij elkaar; soms is er groot gat tussen twee opeenvolgende priemgetallen. Hoe eenzaam kunnen priemgetallen eigenlijk zijn?

We definiëren eerst eenzaamheid voor priemgetallen: Stel we hebben priemgetal *x;* het priemgetal voorafgaand aan *x* noemen we *w*; het priemgetal volgende op *x* noemen we *y*. De afstand tot het voorgaande priemgetal is *(x–w);* de afstand tot het volgende priemgetal is *(y-x).* Nu nemen we als maat voor eenzaamheid de kleinste van *(x-w)* en *(y-x)*. Hoe groter dit getal, des te groter is de mate van eenzaamheid. Er zijn ook andere definities mogelijk maar voor het PrP programma houden we de hier beschreven definitie aan.
Als de *zoekruimte* begint bij nul dan kunnen we voor het tweede priemgetal de mate van eenzaamheid bepalen. Dit is dan tot en met het tweede priemgetal het eenzaamste priemgetal. Voor elk volgend priemgetal bepalen we ook de mate van eenzaamheid en wanneer die mate van eenzaamheid groter is dan die van het vorige eenzaamste priemgetal, dan is het nieuwe priemgetal het nieuwe eenzaamste priemgetal.
Als de *zoekruimte* niet bij nul begint, dan is een gevonden eenzaamste priemgetal niet absoluut de eenzaamste, maar de eenzaamste binnen de verzameling priemgetallen van de zoekruimte.

Bij gesegmenteerde *zoekruimte* zal in het eerste segment snel opeenvolgende eenzaamste priemgetallen gevonden worden. Als er in een volgend segment geen priemgetal wordt gevonden dat eenzamer is dan het eenzaamste in het vorig segment, dan wordt alleen het aantal priemgetallen gerapporteerd in dat segment.

Actie 6. Bereken eenzaamste priemtweelingen

Priemtweelingen zijn nooit echt eenzaam want ze hebben elkaar. Maar we definiëren eenzaamheid voor een priemtweeling als de kleinste afstand van de priemtweeling tot het eerst vorige of het eerst volgende priemgetal.

Stel we hebben een priemtweeling bestaande uit het eerste priemgetal *x;* en het tweede priemgetal *(x+2).* Het priemgetal voorafgaand aan *x* noemen we *w*; het priemgetal volgende op *(x+2)* noemen we *y*. De afstand tot het voorgaande priemgetal is *(x–w);* de afstand tot het volgende priemgetal is *(y-x-2).* Nu nemen we als maat voor eenzaamheid de kleinste van *(x-w)* en *(y-x-2)*. Hoe groter dit getal, des te groter is de mate van eenzaamheid.
Als de *zoekruimte* begint bij nul dan kunnen we voor de eerste priemtweeling de mate van eenzaamheid bepalen. Dit is dan tot de volgende priemtweeling de eenzaamste priemtweeling. Voor elk volgende priemtweeling bepalen we ook de mate van eenzaamheid en wanneer die mate van eenzaamheid groter is dan die van de vorige eenzaamste priemtweeling, dan is de nieuwe priemtweeling

de nieuwe eenzaamste priemtweeling.
Als de *zoekruimte* niet bij nul begint, dan is de eerst gevonden eenzaamste priemtweeling niet absoluut de eenzaamste, maar de eenzaamste vanaf het begin van de *zoekruimte*.

Bij gesegmenteerde *zoekruimte* zal in het eerste segment snel opeenvolgende eenzaamste priemtweelingen gevonden worden. Als er in een volgend segment geen priemtweeling wordt gevonden dat eenzamer is dan de eenzaamste in het vorig segment, dan wordt alleen het aantal priemgetallen gerapporteerd in dat segment.

Vervallen is Actie 7. Bereken palindroom priemgetallen; daarvoor in de plaats komt:

Actie 7. Priem k-tuple clusters (constellaties) opsporen

Een priem k-tupel *(Engels: prime k-tuple)* is een patroon van achtereenvolgende priemgetallen. Het patroon wordt gedefinieerd door *k* getallen die de intervallen tussen een basis priemgetal *p* en de opeenvolgende priemgetallen voorstelt en dat wordt afgebeeld als (n_1, n_2, n_3,... n_k). Hierin is de eerste term gewoonlijk 0 en de volgende termen het interval tussen het basis priemgetal en de volgende priemgetallen. Dus basis priemgetal p = p + n_1, (n_1 = 0) ; de volgende priemgetallen zijn: p+n_2; p+n_3; ...p+n_k. Het zoeken is dus naar basis priemgetallen *p* zodanig dat p+n_2, p+n_3,... p+n_k allemaal priemgetallen zijn. Zo is de k-tuple (0,2) het patroon voor priemtweelingen.
Omdat alle priemgetallen (behalve 2) oneven getallen zijn vormen de intervallen tussen twee opeenvolgende priemgetallen steeds een even getal. Dus n_1,n_2,..n_k moeten even getallen zijn. Verder moeten de termen van de k-tuple in oplopende waarden worden gespecificeerd.
Niet alle waarden vormen een “geldige” prime k-tuple. Voor “geldige” prime k-tuples geldt het ”*Priem k-tuple vermoeden”*, dat er oneindig veel basis priemgetallen zijn waarbij de (k – 1) volgende getallen ook allen priemgetallen zijn. Deze k-tuples worden *admissible* genoemd. De set van *k* achtereenvolgende priemgetallen die voldoet aan het patroon van een k-tuple wordt in dit document en in het PrP programma een “priem cluster” genoemd. De term “priem constellatie” wordt soms ook gebruikt, maar de betekenis van die term kan enigszins variëren, afhankelijk van het artikel of document, waarin de term gebruikt wordt.
De niet “geldige” k-tuples worden *inadmissible* genoemd. Deze kunnen géén, één, of slechts een beperkt aantal, geldige sets van priemgetallen vormen. Bijvoorbeeld de prime k-tuple (0,2,4) is een *inadmissible k-tuple*, omdat er geen

sets van drie priemgetallen bestaan die telkens twee van elkaar verschillen, met uitzondering van (3,5,7). Bij grotere priemgetallen zal, indien het eerste priemgetal bij deling door drie een rest van 1 geeft; noodzakelijkerwijs het tweede deelbaar zijn door drie. Indien het eerste getal bij deling door drie een rest van 2 geeft, dan is het derde getal noodzakelijkerwijs deelbaar door 3.

Nadat de zoekruimte is ingesteld en actie 7 is gekozen wordt een nieuw venster getoond waarin de priem k-tuple kan worden ingevoerd. Het eerste getal is een nul en wordt automatisch ingevuld. Vervolgens worden getallen op volgorde van grootte ingevoerd, telkens gescheiden door een komma. (De C-toets wordt gebruikt als de komma-toets). De maximale grootte van een getal is 98. Er kunnen maximaal 10 getallen worden ingevoerd; alle getallen moeten even zijn. Wanneer de priem k-tuple juist is ingevoerd kan op de OK toets worden gedrukt. Het zoekproces gaat dan van start. De gevonden priem clusters worden vertoond; als eerste het begin priemgetal (wordt basis genoemd); daarna volgen de overige priemgetallen die telkens een waarde groter zijn dan het basis priemgetal: achtereenvolgens de waarden van de termen uit de k-tuple tot het k-de element.
Alle gevonden clusters in de *zoekruimte* worden getoond. Clusters die beginnen vóór het begin van *de zoekruimte* worden niet getoond; ook clusters die eindigen ná het einde de *zoekruimte* worden niet getoond. Indien er twee (of meer) clusters zijn die elkaar overlappen, dan worden beide (of alle) clusters getoond en worden beide of allen als priem-cluster geteld. Een voorbeeld van overlappende clusters is k-tuple (0,6,12) voor basis priemgetal p=251 (251,257,263) en p=257 (257,263,269).
Het aantal priemclusters per segment wordt gerapporteerd, evenals het totaal van de gehele *zoekruimte.*

Zie voor Engelse beschrijving priem k-tuple:
https://en.wikipedia.org/wiki/Prime_k-tuple

Het logbestand
Bij de start van het programma wordt een logbestand op schijf geopend.
Dit bestand krijgt een naam die datum en tijdstip bevat van het moment dat het programma is gestart als volgt: DDMMUUmm.LOG.
Hierin is DD de dag en MM de maand; UU is het uur en mm is de minuut.
Indien het programma tweemaal direct achter elkaar wordt opgestart in dezelfde map dan bestaat het risico dat het tweede logbestand dezelfde naam zou krijgen als het reeds aanwezige logbestand. In dat geval mislukt het openen van het tweede logbestand. Daarvan wordt geen melding gemaakt in de log.