BAB II
DASAR TEORI

2.1 Komputasi Paralel

Komputasi adalah proses yang dijalankan prosesor dalam sebuah rentang satuan waktu yang meliputi:

1) operasi baca: menerima masukan berupa data dengan ukuran tertentu,
2) operasi penghitungan: melakukan operasi aritmatik atau logik terhadap masukan,
3) operasi tulis: mengembalikan hasil penghitungan dengan ukuran tertentu.

Komputasi paralel adalah eksekusi proses pengoperasian aritmatik atau logik yang sejenis secara bersamaan. (Akl, Reif: 1-3)

Sebuah masalah komputasi yang kompleks dibagi menjadi bagian-bagian yang lebih kecil, sederhana, dan dijalankan di beberapa prosesor secara bersamaan sehingga keseluruhan proses dapat diselesaikan dengan lebih cepat.

Pada komputasi sekuensial (serial) masukan diterima dan keluaran dikirim melalui memori internal atau perangkat antarmuka. Sementara pada komputasi paralel sebuah prosesor dapat menerima masukan atau mengembalikan keluarannya melalui *shared memory* atau prosesor lain melalui bus interkoneksi antar prosesor atau jaringan komputer.

Tujuan utama komputasi paralel adalah peningkatan kecepatan komputasi. Hukum Amdahl (Gene Amdahl, 1967) dan Gustafon (John Gustafon, 1988) menjelaskan peningkatan kecepatan komputasi paralel pada sejumlah prosesor identik.

Hukum Amdahl menjelaskan bahwa peningkatan kecepatan komputasi paralel adalah:

$$S = \frac{1}{\frac{1-\alpha}{P} + \alpha} \qquad (2\text{-}1)$$

dengan

α = rasio bagian rutin program yang tidak dapat diparalelisasi terhadap keseluruhan rutin program

P = cacah prosesor.

Hukum Gustafon menjelaskan peningkatan kecepatan komputasi beberapa prosesor dengan persamaan yang lebih sederhana:

$$S=\alpha+P\left(1-\alpha\right) \quad (2\text{-}2)$$

Peningkatan kecepatan komputasi paralel secara teori dibatasi kendala sekuensial bahwa sebuah rutin tidak dapat dipecah menjadi unit yang lebih kecil dan harus dijalankan secara sekuensial di satu prosesor. Sementara secara praktek peningkatan kecepatan komputasi paralel dibatasi performa perangkat keras, penjadwalan proses sistem operasi, konfigurasi lingkungan komputasi dan *overhead* rutin program karena komunikasi dan sinkronisasi antar *proses*.

Program yang dibuat untuk komputasi paralel memiliki kerumitan tersendiri dibanding program sekuensial karena memunculkan kendala baru karena kondisi pacu dan dependensi data antar *proses*. Kondisi pacu dapat ditangani dengan mekanisme lokalisasi dan *locking* data, sementara dependensi data ditangani dengan mekanisme komunikasi *message* antar *proses*. Kedua mekanisme tersebut membutuhkan sinkronisasi antar *proses* yang memunculkan tambahan rutin komputasi yang memperlambat dan menambah kerumitan program.

2.2 *Cluster* Beowulf

Klaster adalah kumpulan elemen yang dapat beroperasi secara mandiri, yang disatukan dengan media untuk menjalankan tindakan yang terkoordinasi dan kooperatif. *Clustering* adalah konsep dan teknik yang digunakan untuk meningkatkan kemampuan dari sistem yang ada melalui agregasi dan sintesis dari elemen-elemen yang lebih sederhana sehingga menciptakan kompleksitas dan bentuk fungsional sistem yang baru.

Cluster (klaster komputer) adalah kumpulan komputer yang dapat beroperasi secara mandiri, yang disatukan dengan jaringan komunikasi data dan mendukung perangkat lunak yang memungkinkan pengaturan rutin beban komputasi secara bersamaan yang bertujuan untuk mengerjakan satu rutin komputasi yang lebih besar.

Di bidang rekayasa komputer, *clustering* diterapkan untuk membuat struktur sistem baru dari elemen-elemen komputasi yang telah ada untuk mendapatkan peningkatan kemampuan komputasi yang jika dilakukan dengan metode konvensional (mengganti perangkat keras dengan performansi yang lebih tinggi) memiliki harga ekonomis yang sangat mahal. *Clustering* termasuk teknik arsitektur sistem komputer yang digunakan untuk meningkatkan performa, kegegasan pengguna, dan reliabilitas.

Cluster komoditas adalah kumpulan *node* komputasi dari komputer *general purpose* dalam jaringan lokal yang perangkat kerasnya dijual secara bebas dan perangkat lunaknya dapat diperoleh dengan bebas. Sehingga *cluster* komoditas memiliki fleksibilitas konfigurasi, *upgrade*, dan kemudahan penggunaan.

Cluster Beowulf adalah *cluster* komoditas yang menggunakan komponen komputer *consumer-grade* dengan harga yang murah (contoh: komponen komputer bekas pakai), menggunakan perangkat lunak gratis dengan lisensi yang bersifat FOSS (*free open source software)* untuk membangun keseluruhan sistem, dan mengakomodasi keperluan komputasi paralel. *Cluster* Beowulf memiliki keuntungan karena stuktur dan metode implementasinya, antara lain: skalabilitas, konvergensi, fleksibilitas konfigurasi, *failover,* kemudahan, kecepatan, dan harga implementasi yang murah dibandingkan sistem *cluster* konvensional.

Komputasi dengan *cluster* Beowulf melibatkan 4 hal dalam pertimbangan sistem:

1. sistem perangkat keras,
2. manajemen sumber daya komputasi,
3. pustaka pemrograman paralel,
4. algoritma paralel.

(Sterling: 15)

Sistem perangkat keras meliputi aspek komponen perangkat keras tiap *node* komputasi, antarmuka dan topologi jaringan. Manajemen sumber daya komputasi adalah serangkaian perangkat lunak yang digunakan untuk instalasi, konfigurasi dan inisialisasi, administasi, dan pemantauan aktivitas, dan perawatan *cluster.* Pustaka pemrograman paralel adalah *framework* yang digunakan untuk membuat program komputasi paralel. Algoritma paralel memberikan model dan pendekatan paralelisasi rutin komputasi.

2.3 Penghitungan Integral Definit dengan Metode Monte Carlo Rerata Sampel

Operasi integral adalah operasi matematik yang menghasilkan nilai pada dimensi yang lebih tinggi dari integrannya sesuai lingkup integrasi yang telah ditentukan, sebagai contoh integral dari persamaan kurva *f(x)* (dimensi satu) akan menghasilkan luasan tertentu (dimensi dua) antara kurva tersebut dan sumbu variabel integrasinya. Integral definit adalah integral dengan batas-batas tertentu untuk tiap-tiap variabel integrasinya.

Integral jamak adalah pengoperasian integral pada beberapa dimensi sekaligus, sebagai contoh integral dari permukaan *f(x,y)* (dimensi dua) akan menghasilkan volume (dimensi tiga) di bawah permukaan tersebut. Integral rangkap tiga terhadap persamaan fungsi pada sistem koordinat kartesian *f(x,y,z)* dengan lingkup integrasi berupa volume akan menghasilkan nilai tertentu pada dimensi keempat, sebagai contoh: massa.

Definisi formal untuk integral definit dimensi satu fungsi *m(x)* dengan batas integrasi antara *a* dan *b* adalah:

$$M = \int_a^b m(x)\,dx \tag{2.4}$$

Pendekatan integral Riemann untuk integral definit dimensi satu tanpa penghalusan kurva (contoh: aturan Simpson dan trapesium) didefinisikan sebagai:

$$M \sim \sum_{i=0}^{N-1} m(x_i)\Delta x_i \tag{2-5}$$

dengan

$$x_i > x_{(i-1)}\,,\ a \le x_i \le b \tag{2-5a}$$

$$\Delta x_i = x_{(i+1)} - x_i \tag{2-5b}$$

Jika Δx_i diseragamkan sepanjang *a* hingga *b*, maka didapatkan pendekatan integral dengan metode segiempat:

$$M \Delta x \sum_{i=0}^{N-1} m(x_i) \tag{2-6}$$

dengan

$$x_i = a + i\frac{(b-a)}{N} \tag{2-6a}$$

$$\Delta x = (b-a)/N \tag{2-6b}$$

Dengan motivasi meningkatkan ketelitian, x_i dapat diubah untuk menghitung nilai *m(x)* di tengah interval, sehingga:

$$x_i = a + (i + 0.5)\frac{(b-a)}{N} \qquad \text{(2-6c)}$$

Dengan penggunaan bilangan acak sebagai pengganti interval reguler Δx dalam Persamaan 2-6, integral definit dimensi satu dengan Monte Carlo rerata sampel didefinisikan sebagai:

$$M \sim \frac{b-a}{N}\sum_{i=1}^{N} m(x_i) \qquad \text{(2-7)}$$

dengan

$$x_i = rand[a..b] \qquad \text{(2-7a)}$$

Definisi formal integral definit yang dilingkupi batas integrasi bangun tiga dimensi dengan volume tertentu adalah:

$$M = \int_V m(v)\,dv \qquad \text{(2-8)}$$

Pada sistem koordinat kartesian, volume integrasi dijabarkan dalam tiga variabel integrasi yang merupakan ketiga sumbu dalam sistem koordinat kartensian.

$$dv = dxdydz \qquad \text{(2-8a)}$$

Volume integrasi dalam penghitungan integral metode balok adalah penjumlahan *cubicle* (balok-balok *kecil* yang saling berhimpitan) sehingga bentuk Persamaan 2-8 dengan *m(x,y,z)* dengan batas integrasi *a* dan *b* untuk variabel integrasi *x*; *c* dan *d* untuk variabel integrasi *y*; *e* dan *f* untuk variabel integrasi *z* adalah:

$$M = \int_e^f \int_c^d \int_a^b m(x,y,z)\,dxdydz \qquad \text{(2-8b)}$$

Dengan pendekatan integral rangkap tiga metode balok (bentuk tiga dimensi dari metode segiempat) didapatkan persamaan dengan ukuran *cubicle* yang konstan adalah:

$$M \Delta v \sum_{i=0,j=0,k=0}^{N-1;O-1;P-1} m(x_i, y_j, z_k) \qquad \text{(2-9)}$$

dengan

$$\Delta x = \frac{(b-a)}{N} \qquad \text{(2-9a)}$$

$$\Delta y = \frac{(d-c)}{O} \tag{2-9b}$$

$$\Delta z = \frac{(f-e)}{P} \tag{2-9c}$$

sehingga ukuran *cubicle* adalah:

$$\Delta v = \frac{(b-a)}{N}\frac{(d-c)}{O}\frac{(f-e)}{P} \tag{2-9d}$$

dengan titik tinjau variabel integrasi yang bergerak secara konstan di masing-masing sumbu tiap iterasi.

$$x_i = a + i(\Delta x) \tag{2-9e}$$

$$y_j = c + j(\Delta y) \tag{2-9f}$$

$$z_k = e + k(\Delta z) \tag{2-9g}$$

Jika nilai-nilai batas integrasi *a, b, c, d, e,* dan *f* berubah antar interasi karena merupakan fungsi *x*, *y*, atau *z;* batas-batas integrasi tersebut dapat dimasukkan dalam interasi. Di samping itu dengan motivasi meningkatkan ketelitian, nilai $x_{i,}\, y_{j,}\, z_k$ dapat ditinjau di tengah *cubicle*-nya (metode *midpoint*). Persamaan pendekatan integral metode balok menjadi:

$$M \sim \sum_{i=0,j=0,k=0}^{N-1;O-1;P-1} m(x_i, y_j, z_k)\Delta v_{ijk} \tag{2-10}$$

dengan

$$\Delta x_{jk} = \frac{(b_{jk}-a_{jk})}{N} \tag{2-10a}$$

$$\Delta y_{ik} = \frac{(d_{ik}-c_{ik})}{O} \tag{2-10b}$$

$$\Delta z_{ij} = \frac{(f_{ij}-e_{ij})}{P} \tag{2-10c}$$

sehingga ukuran *cubicle* adalah*:*

$$\Delta v_{ijk} = \frac{(b_{jk}-a_{jk})}{N}\frac{(d_{ik}-c_{ik})}{O}\frac{(f_{ij}-e_{ij})}{P} \tag{2-10d}$$

dengan titik tinjau variabel integrasi yang bergerak secara konstan dengan interval sesuai pembagian batas variabel integrasinya masing-masing.

$$x_i = a_{jk} + (i + 0.5\ )\Delta x_{jk} \tag{2-10e}$$

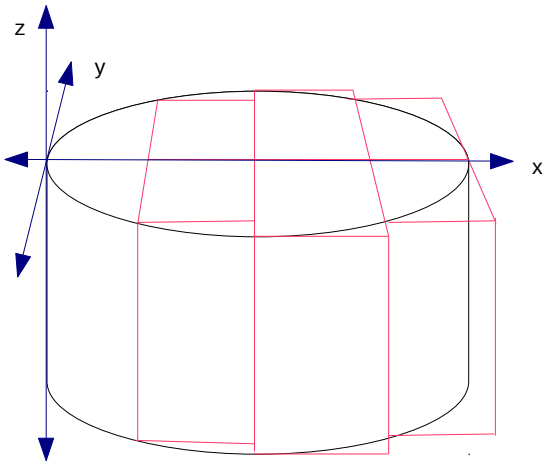
$$y_j = c_{ik} + (j + 0.5)\Delta y_{ik} \quad \text{(2-10f)}$$

$$z_k = e_{ij} + (k + 0.5)\Delta z_{ij} \quad \text{(2-10g)}$$

Hal ini menyebabkan ukuran *cubicle* berbeda-beda karena jarak antara batas integrasi atas dan bawah untuk tiap variabel integrasi di tiap interasi dapat berubah-ubah serta adanya pembagian jarak tersebut dengan konstanta tertentu (*N, O, P).*

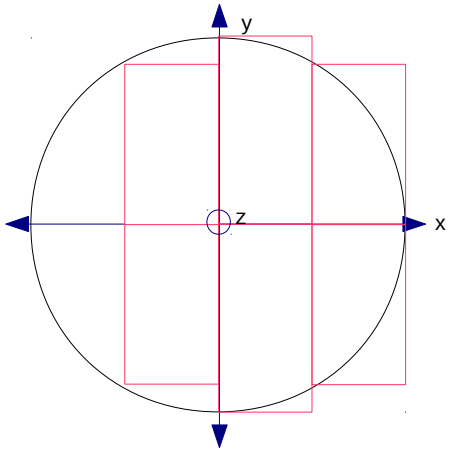
Posisi *cubicle* tidak akan saling tumpang-tindih karena variabel iterasi *i, j, k* merupakan bilangan bulat sehingga batas *cubicle* saling berhimpitan satu sama lain walaupun ukuran *cubicle* yang berbeda-beda antar variabel interasi berbeda. Titik tinjau akan bergerak dengan interval yang konstan sesuai variabel iterasinya terhadap variabel integrasi pasangannya, misal: *x* dan *i.*

Gambar 2.3.1a dan Gambar 2.3.1b memperlihatkan distribusi *cubicle* pada silinder dengan *N=4, O=2, P=1* untuk penghitungan integral metode balok.



Gambar 2.3.1a

Gambar Perspektif Distribusi *Cubicle* pada Silinder dengan Metode Balok



Gambar 2.3.1b

Gambar Tampak Atas Distribusi *Cubicle* pada Silinder dengan Metode Balok

Paralelisasi komputasi integral rangkap tiga metode balok dapat dilakukan dengan membagi batas integrasi yang bukan merupakan fungsi dari variabel lain menjadi beberapa bagian yang masing-masing dikerjakan *proses* terpisah. Hasil integral adalah jumlah hasil dari semua proses.

Penggunaan bilangan acak sebagai untuk menentukan nilai *x, y,* dan *z* dalam jangkauan batas-batas integrasi yang bersesuaian dari Persamaan 2-10 akan menghasilkan definisi formal integral definit dimensi tiga dengan metode Monte Carlo rerata sampel:

$$M \sim \frac{(f-e)(d-c)(b-a)}{N} \sum_{i=0}^{N-1} m(x_i, y_i, z_i) \tag{2-11}$$

dengan masing-masing *x, y,* dan *z* tiap iterasi didapatkan dari bilangan acak antara batas minimum dan maksimumnya masing-masing:

$$x_i = rand[a..b\ ] \tag{2-11a}$$

$$y_i = rand[c..d\ ] \tag{2-11b}$$

$$z_i = rand[e..f\ ] \tag{2-11c}$$

dengan ukuran *cuboid* sebanding dengan volume integrasi.

$$v = (f-e)(d-c)(b-a) \tag{2-11d}$$

Jika nilai-nilai batas *a*, *b*, *c*, *d*, *e*, dan f berubah antar interasi karena merupakan fungsi *x*, *y*, atau *z*, batas-batas dapat dimasukkan dalam interasi sehingga definisi formal integral definit dimensi tiga dengan metode Monte Carlo rerata sampel menjadi:

$$M \sim \frac{1}{N} \sum_{i=1}^{N} m(x_i, y_i, z_i)\left[(f_i - e_i)(d_i - c_i)(b_i - a_i)\right] \tag{2-12}$$

dengan *x, y, z* tiap iterasi diambil secara acak dalam jangkauan batas-batas iterasi sebelumnya.

$$x_{(i+1)} = rand[a_i..b_i] \tag{2-12a}$$

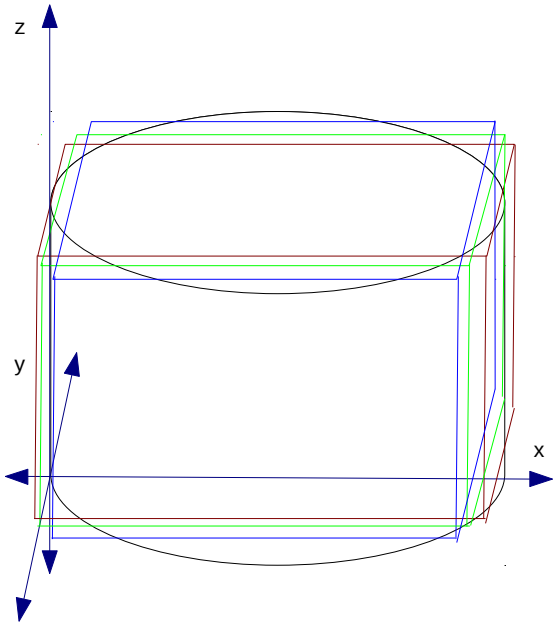
$$y_{(i+1)} = rand[c_i..d_i] \tag{2-12b}$$

$$z_{(i+1)} = rand[e_i..f_i] \tag{2-12c}$$

sehingga ukuran *cuboid* dapat berubah-ubah antar interasi namun tetap sebanding dengan volume integrasi.

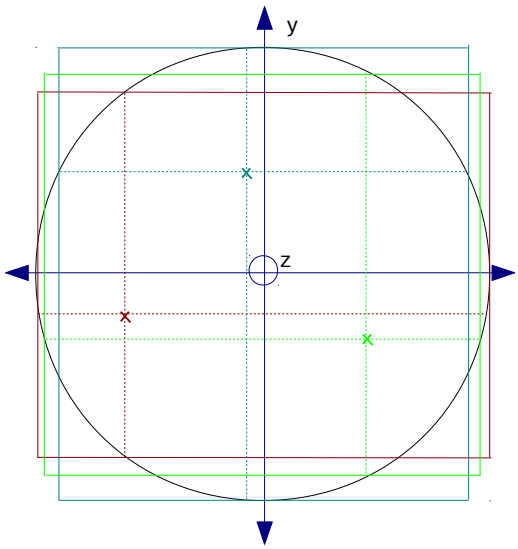
$$v = (f_i - e_i)(d_i - c_i)(b_i - a_i) \tag{2-11d}$$

Gambar 2.3.2a dan Gambar 2.3.2b memperlihatkan distribusi *cuboid* pada penghitungan integral metode Monte Carlo di tiap iterasi yang diwakili warna berbeda.



Gambar 2.3.2a

Gambar Perspektif Distribusi *Cuboid* pada Silinder dengan Metode Monte Carlo



Gambar 2.3.2b

Gambar Tampak Atas Distribusi *Cuboid* pada Silinder dengan Metode Monte Carlo

Paralelisasi komputasi integral definit dimensi tiga metode Monte Carlo dilakukan dengan menghitung *M* di *n proses* komputasi dengan masing-masing *proses* memroses *N* sampel, maka rerata hasil komputasi merupakan *M* dengan cacah *n.N* sampel.

Komputasi integral dengan metode balok dan Monte Carlo rerata sampel memiliki kompleksitas algoritma yang linear $O(n)$ karena tiap sampel hanya dievaluasi sekali sepanjang proses komputasi sehingga waktu komputasi akan meningkat secara linear terhadap cacah sampel yang dievaluasi.

Metode Monte Carlo varian rerata sampel adalah metode numerik dilakukan dengan hanya menghitung rerata sampel dalam cacah yang besar sehingga dengan mudah dapat diimplementasikan pada sistem komputer paralel. Metode Monte Carlo diaplikasikan pada integrasi multi-dimensional dan simulasi stokasik. Metode Monte Carlo memberikan perkiraan pendekatan integral dengan peningkatan ketelitian (akurasi) seiring peningkatan cacah sampel. (Nakano: 1)

Tiap *proses* dapat menyelesaikan iterasi pemrosesan himpunan sampelnya masing-masing tanpa komunikasi yang intensif dengan *proses* lain, maka algoritma komputasi paralel integral definit dimensi tiga metode balok dan metode Monte Carlo rerata sampel dapat diimplementasikan secara *pleasingly parallel.*

2.4 Open MPI 1.7

Komputasi paralel di sebuah *cluster* Beowulf dilakukan dengan membagi proses komputasi menjadi bangian-bagian dan menggunakan beberapa *proses* yang masing-masing dijalankan di prosesor berbeda. Umumnya sebuah program yang sama dapat digunakan di semua *proses*, namun dengan masukan dan parameter yang berbeda.

Pada permasalahan yang sederhana, program dapat digunakan beberapa *proses* berbeda hanya dengan masukan dan parameter yang berbeda tanpa komunikasi antar proses. Namun jika permasalahan menjadi kompleks, komunikasi antar *proses* diperlukan dan perlu didesain dengan mangkus dan sangkil pada tahap implementasi.

Salah satu pendekatan komunikasi antar proses adalah mengoordinasikan aktivitas *proses-proses* dengan mekanisme pengiriman dan penerimaan *message* berupa deretan *byte* dengan struktur data tertentu antar *proses.* Spefisikasi mekanisme komputasi antar proses dengan *message* ini disebut MPI.

MPI (*message-passing interface)* adalah spesifikasi putsaka *message-passing.* Terdapat 3 bagian utama dari deskripsi spesifikasi MPI:

- MPI menjabarkan model *message-passing* komputasi paralel yang mana tiap proses memiliki lingkup alamatnya masing-masing dan melakukan sinkronisasi antar proses dengan menerima dan mengirimkan *message*,
- MPI menjabarkan spesifikasi pustaka: kumpulan sub-rutin dan argumennya. MPI bukan merupakan pustaka dalam bahasa pemrograman tertentu, namun rutin MPI diimpelemtasikan melalui bahasa pemrograman konvensional antara lain: C, C++, dan Fortran,
- MPI adalah spesifikasi, bukan merupakan implementasi pemrograman. Spesifikasi MPI dibuat oleh MPI Forum. Terdapat tiga spesifikasi umum MPI: MPI-1 (1994-1998), MPI-2 (1998-2009), dan MPI-3 (2012).

OpenMPI 1.7 memiliki implementasi lengkap dari spesifikasi MPI-2.1, sebagian besar spesifikasi MPI-3.0, dan sebagian kecil spesifikasi MPI-1 (terutama untuk antarmuka bahasa pemrograman Fortran) yang secara fungsional telah banyak digantikan spesifikasi MPI-2.

Dalam implementasinya, Open MPI 1.7 menggunakan *suite* SSH (Secure Shell) untuk menjalankan program secara *remote* di komputer lain.

2.5 Pembangkit bilangan acak semu - random, srandom, rand, srand.

Fungsi random() menggunakan pembangkit bilangan acak dengan umpan balik tambahan non-linear dengan tabel standar yang terdiri dari integer dengan panjang 31 bit untuk memberikan nilai kembalian bilangan acak semu berurutan dengan jangkauan 0 hingga RAND_MAX. Di sistem operasi GNU/Linux 3.x dengan glibc 2.15, RAND_MAX bernilai 2147483647 (2^{31}) dengan periode pembangkit bilangan acak adalah $16*(2^{32}-1)$.

Fungsi srandom() mengatur argumennya sebagai *seed* untuk membangkitkan deretan bilangan acak semu yang dihasilkan fungsi random(). Deretan bilangan acak semu ini dapat diulangi dengan memanggil fungsi srandom() kembali dengan nilai *seed* yang sama. Jika nilai *seed* tidak diatur secara eksplisit, fungsi random() secara otomatis diberikan *seed* dengan nilai 1.

Fungsi rand() mengembalikan integer acak semu dengan jangkauan inklusif 0 hingga RAND_MAX. Fungsi srand() mengatur argumennya sebagai *seed* untuk membangkitkan deretan bilangan acak semu yang dihasilkan fungsi rand(). Deretan bilangan acak semu ini dapat diulangi dengan memanggil fungsi srand() dengan nilai *seed* yang sama. Jika tidak ada nilai *seed* yang diberikan, fungsi rand() akan secara otomatis diberi nilai *seed* 1.

Fungsi rand() dan srand() pada pustaka standar C Linux menggunakan pembangkit bilangan acak yang scama dengan fungsi random() dan srandom().

Pembangkit bilangan acak standar sistem operasi GNU/Linux 3.x mempunyai distribusi bilangan yang bersifat diskret dan *uniform.*