CSIM (10-04-07)
Generazione di numeri casuali e controllo della simulazione

Emiliano Casalicchio
emiliano.casalicchio@uniroma2.it

Agenda

- Generazione di numeri casuali in CSIM
- Terminazione della simulazione in CSIM

Random Numbers

- Possibilità di generare
  - Single stream
  - Multiple stream
- Ditribuzioni Continue
- Distribuzioni Discrete
- Distribuzioni Empiriche

Continuous Distributions

- double uniform(double min, double max)
- double triangular(double min, double max, double mode)
- double beta(double min, double max, double shape1, double shape2)
- double exponential(double mean)
- double gamma(double mean, double stddev)
- double erlang(double mean, double var)
- double hyperx(double mean, double var)
- double weibull(double shape, double scale)
- double normal(double mean, double stddev)
- double lognormal(double mean, double stddev)
- double cauchy(double alpha, double beta)
- double hypoexponential(double mn, double var)
- double pareto(long a)
- double zipf(long n)
- double zipf_sum(long n, double *sum)

Discrete Distributions

- long random_int(long min, long max)
- long bernoulli(double prob_success)
- long binomial(double prob_success, long
- num_trials)
- long geometric(double prob_success)
- long negative_binomial(long success_num, double prob_success)
- long poisson(double mean)

Empirical Distributions

- void setup_empirical(long n, double prob[], double cutoff[], long alias[])
  - Deve essere invocata in fase di setup dei generatori di numeri casuali
  - prob[] contiene le probabilità di generare un numero (es: prob=[0.3, 0.13, 0.17, 0.2, 0.2]
- double empirical(long n, double cutoff[], long alias[], double value[])
  - Permette di estrarre un valore dal set value[] in base alle probabilità specificate in prob[]
  - value[] contiene i numeri da generare (es: value=[3, 4, 10, 12, 17])
- prob[i]= probabilità di estrarre il valore value[i]

Single stream

- Di default CSIM usa un'unico stream di numeri, ed il seme è automaticamente impostato ad 1
- Per cambiare seme:
  - void reseed(STREAM s, long n)
  - Es: reseed(NIL, 13579);
  - Il seme può essere cambiato in qualsiasi punto della simulazione
- Per conoscere lo stato attuale di uno stream
  - long stream_state(STREAM s)
  - Es: stream_state(NIL) ritorna lo stato della stream unica.
  - Es: reseed(NIL, 13579); i=stream_state(NIL) /* i=13579 */

Multiple stream

- Variabili aleatorie indipendenti richiedono sequenze indipendenti. Utililizzando stream separate si è sicuri di avere stream indipendenti
- CSIM mette a disposizione 100 stream indipendenti i cui seed sono spaziati di 100.000 valori.
  - STREAM create_stream(void)
  - Es:

```
STREAM s;
s = create_stream();
reseed(STREAM s, long n);
```

- double stream_<distribFuncName>(STREAM s, <distrib_parameters>);

Esempi CSIM

- single-stream.c, multistream.c
- single-stream.c
  - Seed=35

```
t=0.011882, srv=2
t=0.11806, srv=2
t=0.615691, srv=3
t=1.27062, srv=4
t=1.72347, srv=1
t=1.8161, srv=2
t=1.83036, srv=3
t=1.86344, srv=0
t=1.93414, srv=3
t=3.95347, srv=4
t=5.74728, srv=4
t=7.05873, srv=0
t=9.43674, srv=3
t=9.68803, srv=2
t=9.81166, srv=3
t=9.90979, srv=4
t=10.3409, srv=0
t=10.7878, srv=2
```

  Seed=37

```
t=3.47511, srv=2
t=3.54453, srv=3
t=4.76106, srv=4
t=5.05914, srv=0
t=5.3492, srv=3
t=6.11876, srv=4
t=6.73165, srv=2
t=6.85498, srv=1
t=8.16424, srv=2
t=11.5625, srv=4
t=11.6885, srv=2
t=12.5472, srv=4
t=12.9105, srv=0
t=14.002, srv=4
t=14.2299, srv=1
t=14.9446, srv=2
t=16.217, srv=1
t=18.1702, srv=3
```

Trace delle simulazioni

- Trace simulation
  - Possibilità di tracciare i processi
    - trace_on(); trace_process(processName);
  - Possibilità di tracciare gli oggetti, e.g. eventi, storages, facility, etc…

Esempio di trace

time	process	id	pri	status
21.382	AT2	4	1	use facility httpd for 0.004
21.382	AT2	4	1	reserve facility httpd
21.382	AT3	5	1	use facility httpd for 0.030
21.382	AT3	5	1	reserve facility httpd
21.382	AT4	6	1	use facility httpd for 0.089
21.382	AT4	6	1	reserve facility httpd
21.382	AT5	7	1	use facility httpd for 0.025
21.382	AT5	7	1	reserve facility httpd
21.382	AT6	8	1	use facility httpd for 0.028
21.382	AT6	8	1	reserve facility httpd
21.382	AT7	9	1	use facility httpd for 0.004
21.382	AT7	9	1	reserve facility httpd
21.382	AT8	10	1	use facility httpd for 0.357
21.382	AT8	10	1	reserve facility httpd
21.382	AT9	11	1	use facility httpd for 0.114
21.382	AT9	11	1	reserve facility httpd
21.382	AT10	12	1	use facility httpd for 0.028
21.382	AT10	12	1	reserve facility httpd
21.382	AT0	33	1	wait event burst (#9d49e28)
21.382	AT1	34	1	wait event burst (#9d49e28)
21.382	AT2	35	1	wait event burst (#9d49e28)
21.382	AT3	36	1	wait event burst (#9d49e28)
21.382	AT4	37	1	wait event burst (#9d49e28)
21.382	AT5	38	1	wait event burst (#9d49e28)
21.382	AT6	39	1	wait event burst (#9d49e28)
21.382	AT7	40	1	wait event burst (#9d49e28)
21.382	AT8	41	1	wait event burst (#9d49e28)
21.382	AT9	42	1	wait event burst (#9d49e28)
21.382	AT10	43	1	wait event burst (#9d49e28)
21.510	WC10	23	1	terminate process
22.535	WC20	44	1	use facility httpd for 1.258
22.535	WC20	44	1	reserve facility httpd
22.548	WC11	24	1	terminate process

Intervallo di confidenza e controllo della lunghezza dei run.

- Possibilità di calcolare l'intervallo di confidenza del valore medio di ogni oggetto per collezionare le statistiche
  - void table_confidence(TABLE t)
  - void qtable_confidence(QTABLE qt)
  - void meter_confidence(METER m)
  - void box_time_confidence(BOX b)
  - void box_number_confidence(BOX b)
- La tecnica utilizzata per calcolare l'intervallo di confidenza è la: *Batch means analysis.*
  - Viene calcolato l'intervallo di confidenza per 3 livelli di accuratezza: 90, 95, 98%

Esempio

- confidenceInterval.c
  - out.confidenceIntervalYesNo
  - out.confidenceIntervalYesYes

Run length control

- E' possibile controllare la terminazione di una simulazione basandosi sull'osservazione di una o più variabili e sulla convergenza del loro valore medio all'interno di un'intervallo di confidenza.
  - void table_run_length(TABLE t, double accuracy, double conf_level, double max_time)
  - **accuracy** specifica il massimo errore relativo che sarà consentito al valore medio della variabile osservata
  - **conf_level** specifica l'intervallo di confidenza, eg=0.95
  - **max_time** specifica il tempo massimo da simulare se la variabile osservata non converge
- Quando la simulazione terminerà, l'evento converged sarà impostato ad occurred
  - wait(converged);
- E' possibile richiedere la convergenza di più varibili.

Esempi

- runLenghtControl.c