University of Ljubljana
Faculty of Computer and Information Science

Modeliranje računalniških omrežij

OMNeT++ Analiza simulacijskih rezultatov

Laboratorijske vaje

Torek, 29. oktobra 2013

Definicija eksperimentov

- eksperimente definiramo v `ini` datotekah
- privzeta datoteka je `omnetpp.ini`

Določanje parametrov (omnetpp.ini)

```
omrezje.modul.parameter = vrednost/izraz
**.modul.parameter = vrednost/izraz
**.parameter = vrednost/izraz
```

Wildcards: `*`,`**`,`?`,`{a-f}`,… – glej Manual str. 192

- le tisti parametri, ki še niso bili ovrednoteni v *ned* datotekah
- več ponovitev z različnimi parametrskimi vrednostmi

```
*.ime_parametra = ${1, 2, 5, 10..50 step 10}
*.ime_parametra = ${N=1, 2, 5, 10..50 step 10}
```

- več ponovitev z omejitvami

```
*.param1 = ${i=1..10 step 2}
*.param2 = ${j=1..20 step 3}
constraint = $j <= sqrt($i)
```

- spreminjanje *seed*-a s parametrom `seed-set` (privzeta vrednost je `${runnumber}`).

Runtime analiza (debug)

- Debug: Run→Debug As→OMNeT++ Simulation…
- `WATCH`:
  - v `initialize()` metodi uporabimo makro `WATCH(ime_spremenljivke);`
  - vrednosti spremljamo preko Tkenv orodja: Inspect Network→dvojni klik na modul
- izpisovanje vrednosti v konzolo orodja Tkenv:

```
EV << niz1 << niz2 << spremenljivka1 << ...;
```

- izpisovanje v Inspect Network okno kot *Tag* modula:

```
char buf[80];
sprintf(buf, "oznaka: %d\n",spremenljivka);
getDisplayString().setTagArg("t",0, buf);
```

Beleženje statistike

- uporaba signalov (*signals)*
- moduli oddajajo signale z določenimi vrednostmi (npr. dolžina čakalne vrste)
- deklaracija signalov v NED datoteki *simple* modula v sekciji `parameters`
- v ned datotekah specificiramo signale, ki jih želimo opazovati in načine zbiranja vrednosti
- v ini datoteki lahko določimo kam, kako in kaj se bo beležilo pri posameznem eksperimentu
- 2 tipa statističnih podatkov
  - skalarji: izračunajo se v finish() metodi, že obdelani podatki
  - vektorji: vsebujejo vse signalizirane podatke

Dodajanje signalov (C++)

- deklaracija (v `private` delu `*.h` datoteke)

```
simsignal_t imeSignala;
simsignal_t queueLengthSignal;
```

- registracija signala (v `initialize()` metodi modula)

```
imeSignala = registerSignal("imeS");
queueLengthSignal =
registerSignal("queueLength");
```

- oddajanje vrednosti ob spremembah:

```
emit(imeSignala, vrednost);
emit(queueLengthSignal,queue.length());
```

Konfiguracija beleženja (NED)

- signali se avtomatsko beležijo, če jih dodamo v NED datoteko modula
- signale dodamo v `parameters` del kot

```
@statistic[imeS](title, unit, interpolationmode, enum, record);
```

- parametri v oklepaju so opcijski, njihov pomen pa je sledeč
  - `title`: niz – ime statistike (uporabi se npr. pri vizualizaciji)
  - `unit`: merska enota
  - `interpolationmode`: način interpolacije, če je ta potrebna
  - `enum`: interpretacija integerjev (npr. `"IDLE=1,BUSY=2"`)
  - `record`: način beleženja podatkov (`vector, count, last, sum, mean, min, max, timeavg, stats, histogram`) – user manual, str. 237

```
@statistic[queueLength]("vrsta",record=vector,mean);
```

Konfiguracija beleženja (ini)

- uporabnik lahko določa, kako, kam in kateri podatki se bodo beležili
- spreminjamo lahko le tiste statistike, ki so bile določene v NED datoteki
- določanje datotek za shranjevanje:

```
output-vector-file = ${resultdir}/${configname}-${runnumber}.vec
output-scalar-file = ${resultdir}/${configname}-${runnumber}.sca
```

- vklop/izklop beleženja:

```
**.scalar-recording = true/false
**.vector-recording = true/false
```

- če ne kličemo metode `finish()`, se skalarji ne izračunajo!

Konfiguracija beleženja (ini)

- način beleženja podatkov

```
**.result-recording-mode = vector, count, last, sum, mean, min, max, timeavg, stats, histogram
```

- intervali beleženja vektorjev

```
**. vector-recording-intervals = 0..100s, 500s..900s, 1100s..
```

- shranjevanje vrednosti parametrov (samo za numerične)

```
**.param-record-as-scalar = true
**.parameter.param-record-as-scalar = true
```

- warm-up (default = 0s)

```
warmup-period = 20s
```

- **določitev časa simuliranja**

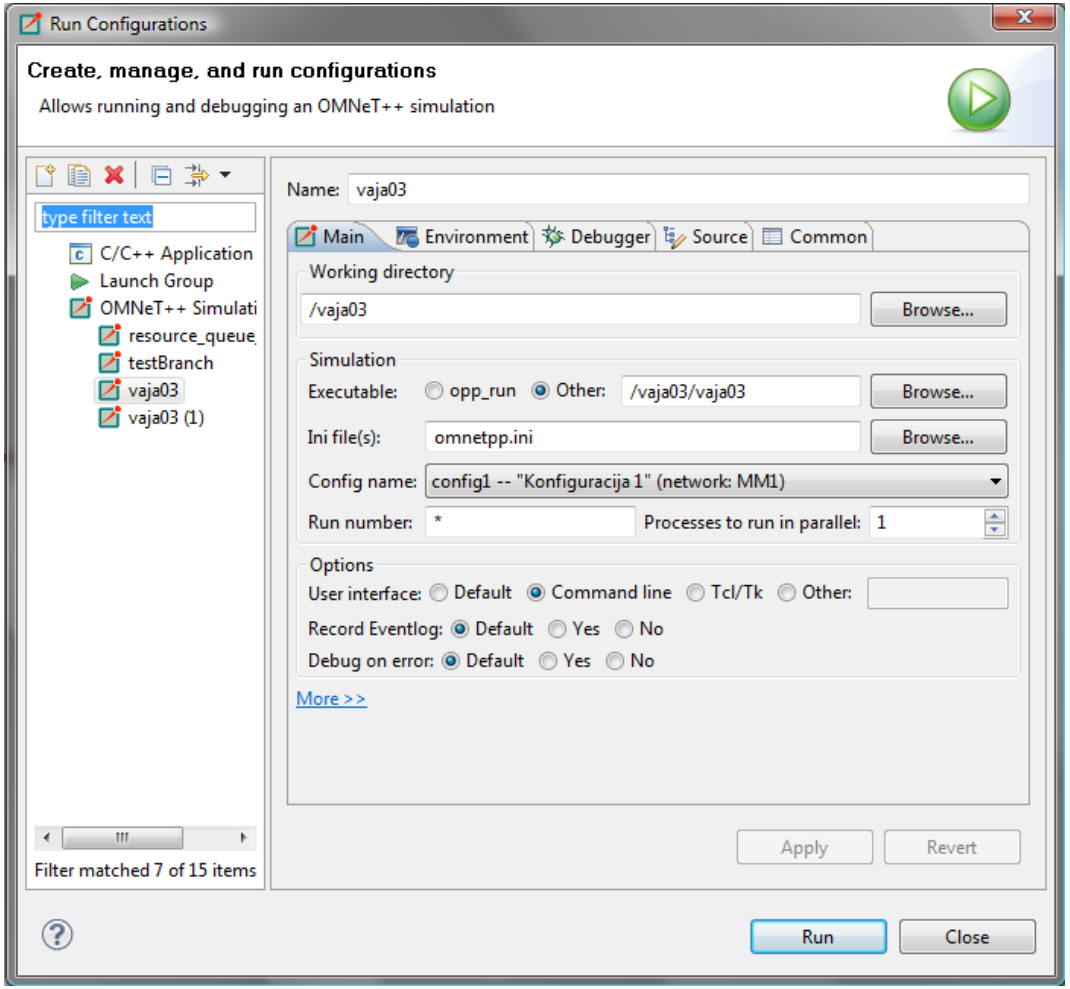
```
sim-time-limit = 10000 s
```

- **vklop beleženja dogodkova** (*eventlog*)

```
record-eventlog = true
```

Pogajanje simulacije

Run→
Run Configurations



Analiza rezultatov

- `*.vec` datoteka: datoteka z vektorskimi podatki
- `*.sca` datoteka: datoteka s skalarji
- `*.elog` datoteka: dnevniška datoteka dogodkov (eventlog datoteka)

- File→New→Analysis File (anf)
- Odpre se *Analysis Editor*
  - *Inputs*: dodamo `*.vec` in `*.sca` datoteke iz direktorija *results*
  - *Browse Data*: pregledovanje podatkov v datotekah
  - *Dataset*: prikazovanje in analiza rezultatov

Dataset

- ustvarimo nov *Dataset*
- dodajanje podatkov – Add
  - določimo *Data Type*
  - določimo *Filter Pattern*: do namigov pridemo s *ctrl+space*
- obdelava podatkov – *Apply* in *Compute*
- prikazovanje podatkov
  - Bar Chart: skalarji
  - Line Chart: vektorji
  - Histogram Chart: histogrami
  - Scatter Chart: za katerekoli podatke

Za podrobnosti glej *IDE User Guide* – poglavje 9.
http://www.omnetpp.org/doc/omnetpp/UserGuide.pdf

Naloga

V modelu M/M/c dodajte beleženje

- faktorja uporabnosti resursov,
- števila zahtev, ki čakajo na proste resurse.

Na podlagi analize simulacijskih rezultatov aproksimirajte **optimalno število resursov**, pri čemer je čas procesiranja ene zahteve enak 10 s, medprihodni časi zahtev pa

a) 3 sekunde,

b) 5 sekund,

c) 7 sekund.

Vsako simulacijo poganjajte 10000 sekund.

Namigi

- opazujte faktor zasedenosti resursov in dolžino čakalne vrste
- faktor zasedenosti mora biti čim večji, pri čemer dolžina čakalne vrste s časom ne sme naraščati v neskončnost
- faktor zasedenosti opazujte kot skalar, dolžino čakalne vrste pa kot vektor