DeSiNe

February 27, 2008

1 Installation

1.1 Quick install

To install DeSiNe, unzip the archive and run `make`.

- `tar xzvf Desine.tar.gz`
- `cd ./Desine`
- `make desine`

The executable is now available as `./obj/desine`.

1.2 Advanced install

Where necessary, modify `src/makefile.global` to add optimization flags or change the compiler (defaults to gcc/g++). The make-file contains several compiler directives:

`-DNO_TIMER`	Omits the calculation of the average running time for a single route computation. Adding this directives improves the performance since the calculation queries the system clock.
`-DNO_TRACES`	Excludes the debugging messages at compilation.
`-DNO_THROUGHPUT`	Excludes data-mining of the throughput measures.
`-DNO_STD_DEV`	Excludes data-collection and computation of the variance of the measurements.

Table 1: Compiler directives.

2 Program arguments

DeSiNe accepts several arguments at the command line. The syntax is as follows:

- `desine <parameter-file> [OPTIONS]`

where the `<parameter-file>` argument is mandatory and should indicate the location of the file containing the simulation parameters (see Section 3). The following options are available:

`--h, --help`	Display a short help message.
`--print`	Print results to screen.
`--V, --version`	Display version and compile date.
`--filename <nm>`	Set base filename/prefix ¡nm¿ for result filenames. Default prefix is `out`.
`--v`	Verbose mode[1].

Table 2: Command-line arguments.

3 Simulation parameters

The simulation parameters are enclosed within a file that is read by DeSiNe. Each line in this file must end with a end-of-line character. Every line starting with a # is treated as a remark and is ignored by the parser. Each token (i.e. key-value pair) must be defined on a separate line and all tokens are case sensitive! The tokens can be placed in random order and several tokens can be defined multiple times (multiplicity, described below). Key and value must be separated by whitespace(s). The tokens that are defined, with their value-types, are displayed in Table 3. The multiplicity field indicates whether a token may be present more than once. This is useful when multiple scenarios are examined in one run, or, in case of the `QoSCons` field, it represents the constraints in a multi-constrained routing scenario.

Table 4 describes how to format the string that describes a random variable. Note that a mixed random variable is simply a weighed combination of multiple other random variables, where the probability that a distribution is used is proportional to the weight.

The algorithms and link cost functions that are available in DeSiNe are shown in Tables 5 and 6. The link-state update policies are presented in Table 7. The source tarball contains examples on how to use the parameters.

References

[1] S. Avallone, F.A. Kuipers, G. Ventre, and P. Van Mieghem. Dynamic routing in QoS-Traffic Engineered networks. In *EUNICE IFIP WG 6.6, WG 6.4 and WG 6.9 Workshop*, pages 222–228, Colmenarejo, Spain, Jul 2005.

[1] If the source is compiled with the traces enabled, hence without the `NO_TRACES` directive, the traces can be made visible in verbose mode.

[2] The string must describe the distribution as described by Table 4.

Key	Value-type	Multiplicity	Description
`Flows`	unsigned int	No	Number of flows that will be generated each iteration
`Warmup`	unsigned int	No	Number of warmup flows when starting up the simulation.
`Iterations`	unsigned int	No	Number of iterations made for this simulation.
`Topology`	string	No	String with the filename describing the topology.
`Logdir`	string	No	String with path to directory where logfiles are placed.
`Endpoints`	`U` or `E`	No	States whether source and destination are uniformly chosen (`U`) or exponentially (`E`).
`Farrival`	string[2]	No	Stochastic distribution for the Inter-arrival time between two successive flow requests.
`Fduration`	string[2]	No	Stochastic distribution for the duration of each flow.
`Fbandwidth`	string[2]	No	Stochastic distribution for the requested bandwidth for each flow.
`QoSCons`	string[2]	Yes	Stochastic distribution for the (additive) QoS constraints. The current limitation on the number of QoS contraints is two.
`Algorithm`	string	Yes	The routing algorithm(s) that is(are) to be invoked during each iteration. Each algorithm that should be applied, must be stated on a separate line. The value is a combination of one of the algorithms from Table 5 and a linkcost function from Table 6.
`LsupdateInfo`	string	Yes	Link state update policy(s) and relative parameters. For each policy, or the same policy with different parameters, there will be iterations using one or more algorithms as given in Algorithm. The value is one of the linkstate update policies shown in Table 7

Table 3: Simulation parameters.

Distribution	Syntax	Example
Uniform	`U <min> <max>`	`U 1 10`
Exponential	`E <mean>`	`E 10`
Mixed	`<N> <var1> <prob1> <var2> <prob2> ...<varN> <probN>`	`2 U 1 10 0.30 E 10 0.70`

Table 4: Syntax of random variable distribution.

[2] B. Lekovic and P. Van Mieghem. Link state update policies for quality of service routing. In *IEEE 8th Symposium on Communications and Vehicular Technology in the Benelux (SCVT)*, pages 123–128, Delft, The Netherlands, October 2001.

Keyword	Description
`NewMira`	New Mira
`MI-PA`	S-Mira
`SP`	Shortest Path
`WSP`	Widest Shortest Path
`SAMCRA`	SAMCRA
`SAMCRA-B`	SAMCRA-B
`SAMCRA-A`	SAMCRA_after
`TE-DB`	TE-DB

Table 5: Routing algorithms available in DeSiNe. See [1] for a description on some of the less known algorithms.

Keyword	Description
`Inverse`	Inverse link capacity
`PowerDot75`	power-function (0.75) of link capacity
`Delta`	Delta
`Delta3`	Delta3
`SquareSin`	Square Sin
`MinDelay`	Minimum delay
`MinHop`	Minimum hop

Table 6: Linkcost functions available in DeSiNe.

Keyword	Arguments	Description
`Moving_Average_Relative`	`<ws> <th>`	Moving average relative threshold based
`Moving_Average_Absolute`	`<ws> <th>`	Moving average absolute threshold based
`Moving_Average_class`	`<ws> <gf> <bb>`	Moving average class based
`Period`	`<pe>`	Periodical time based
`Holddown_Timer_class`	`<pe> <gf> <bb>`	Holddown timer class based
`Holddown_Timer_threshold`	`<pe> <th>`	Holddown timer threshold based

Table 7: Linkcost functions available in DeSiNe. `ws` stands for window size. `gf` stands for growth factor. `th` stands for threshold. `bb` stands for bandwidth base. `pe` stands for period. Detailed information on the link-state update policies can be found in [2].