New SNE Method for the Space Extractor

S. de Graaf

Circuits and Systems Group
Faculty of Electrical Engineering,
Mathematics and Computer Science
Delft University of Technology
The Netherlands

Report EWI-ENS 06-03
Nov 20, 2006

Copyright © 2006 by the author.
All rights reserved.

Last revision: Dec 15, 2006.

1. INTRODUCTION

This report describes the implementation of a new SNE method into the *space* program, the layout to circuit extractor.

SNE stands for Selective Node Elimination. By SNE, the nodes of the extracted circuit are eliminated based on there weight. The weight of a node is calculated and depends on the chosen SNE frequency and the connected RC elements to the node. Only nodes below the critical weight (the SNE tolerance) are eliminated. However, terminal nodes and nodes of device pins are never eliminated.

This report represent the test results of the new implemented SNE method. For the tests are used the two example cells, which were used by Peter Elias (see paper "Extracting Circuit Models for Large RC Interconnections that are Accurate up to a Predefined Signal Frequency").

The new SNE method calculates the node weight with the formula:

```
    weight = 2 * π * frequency * Σ(C) / Σ(G)
```

To calculate the above formula, the program only needs to look to the connected capacitors and resistors of the node. This calculation can therefor performed much faster than by the other SNE methods. The other SNE methods calculates the node weights based on there elimination and must therefor look to all neighbor nodes.

Normally, the SNE methods can only be used in combination with the moments node elimination method. However, the new implemented method can also be used with the standard (gaussian) node elimination method of the *space* program.

2. HOW TO USE THE NEW SNE METHOD

To use a SNE method, you must give option -G to the *space* program. To use the new method, you must also set parameter "sne.norm" to 3. This can be done in the *space* parameter file, but also with option -S on the command line:

```
    % space -rCG -Ssne.norm=3 testcellname
```

The new SNE method uses default the "moments.max=0" parameter setting. This disables the moments node elimination method. Note that the other SNE methods (sne.norm < 3) are using default "moments.max=2".

Note that nodes without capacitors and/or resistors cannot selectively be eliminated. These nodes get a weight equal to zero and are always eliminated.

Note that nodes, which are eliminated, are eliminated in a certain priority order. Therefor, the *space* program has a node priority queue. The nodes with lowest priority (degree) are first eliminated. The index of a node in this priority queue is called the degree of the node. The lowest possible degree for a node is zero. Which means, that nothing is connected to the node.
Default, the degree is calculated with the formula:

```
    degree = node -> res_cnt + node -> cap_cnt
```

However, when the SNE method is used, two or more queues are used. At least two, to separate the nodes with a weight ≥ sne.tolerance from the delayed nodes with a weight < sne.tolerance. With parameter sne.resolution you can choice for more weight queues. This is important for the order in which nodes are eliminated. Because, when nodes have the same degree and different weights, the program shall else possible eliminate the wrong internal node first. And the internal node, which gets a weight ≥ sne.tolerance and stays, is maybe not the node which you want to have. This gives also a network, which you have not expected. This problem happens with the "rc2" test cell by certain sne.frequency.

3. IMPORTANT SPACE PARAMETERS

3.1 Node queue parameters

The following node priority queue parameters can be set:

elim_order	With this parameter, it is possible to choice for another node degree calculation method. The node degree (an integer value) is used as an index for the node priority queue, where to store the node. Default (value 0), the node degree is the sum of the connected resistors and capacitors. Three other values can be used. Use a value of 1, when only the node resistor count must be used. Note that another choice may result in another elimination order and another extracted network.
max_delayed	This parameter can be used to change the maximum number of nodes (default 500), which may be stored in the priority queue. When the maximum is reached and a new delayed node must be stored, then this node forces an elimination of one delayed node with lowest degree from the queue. Note that this eliminated node can be a node of another (maybe not ready) node group. When this parameter is changed, then possible also the elimination order of nodes is changed, and this can result in a different extracted network. Note that by the SNE method, delayed nodes with a weight $\geq$ sne.tolerance are not counted.

3.2 SNE parameters

Put the string "sne." before the parameter or use a "begin sne" block.

frequency	The frequency used for the SNE of nodes. This value must be > 0 (default 1e9 Hz), else SNE is disabled.
norm	This parameter value must be ≥ 0 and ≤ 3 (default 0). Three sne.norm values are possible for the old SNE method. The value 3 is special, because it is used to enable the new SNE method. For the old SNE method, default (value 0) uses the L-infinity norm to calculate the branch weight. By value 1 the L-1 norm is used and value 2 uses the L-2 norm.
errorfunc	This parameter value must be ≥ 0 and ≤ 3 (default 0). The parameter is only used for the old SNE method. When the branch weight for a node is calculated a certain error criterium is used.
fullgraph	The calculated weight value is based on the branches which are evaluated. This parameter is not used for the new SNE method. This parameter value must be ≥ 0 and ≤ 2 (default 0). By the default setting, only the grounded branches are used. By a value of 1, only

	the capacitive branches are used. And by a value of 2, all branches are used.
tolerance	This is a more important parameter for setting the critical weight level. The default value of 0.05 is in most cases acceptable (the value must be > 0). All nodes with a weight below the sne.tolerance setting are eliminated.
resolution	With this parameter you can choice for more node weight queues. Default, only one weight queue is used (the value must be >= 1). The delayed nodes in the highest weight queue are not eliminated. For nodes with the same degree, a better choice can be made, which first must be eliminated when there are more separate queues.
print_elimcount	Default "off". Prints info about each delayed node (a number, the node weight and a comment).

3.3 MOMENTS parameters

Put the string "moments." before the parameter or use a "begin moments" block.

max	This parameter sets the maximum number of moments used. The default value is 2 (0 for sne.norm is 3 setting). A default value of 2 gives one extra moment. This extra moment is stored in the capacitor element (for ground/substrate caps in the node). The moment 0 value is the value of a resistor element. The moment 1 value is the value of a capacitor element. This parameter must be ≥ 0 and ≤ 10. For the old SNE method the value must be ≥ 2. For the new SNE method, this value can be set to use the moments node elimination method.
out_rc	Default "off". When "on", function outputSeriesRC is done.
out_l	Default "off". When "on", function outputSeriesL is done. Negative capacitance values are outputted as L's.
out_negc	Default "on" (note: "off", when out_l is "on"). When "off", do not output capacitors with negative value.
out_negr	Default "on". When "off", do not output resistors with negative value.
print	Default "on" (note: "off", when out_l is "on" or sne.max < 1). Print the moments values in the attribute list of each outputted capacitor.

4. EXAMPLE CELL RC2

The extraction results depends on the chosen extraction method (-rC, -r3C, -z3C). And, of coarse, of the used technology file and parameters file (see appendix B and C). Note that the parameters for reduction heuristics are not used. Only data for the used masks np (polysilicon) and nm (metal) are given. Note that the metal conductor has a low sheet res value (< 1 ohm). For the masks we use the "nmos" process and a lambda of 1 micron.

Now follow two 2D capacitance extraction results (the second also with lateral caps):

```
% space3d -Espace.def.t -Pspace.def.p -rC RC2
% xsls RC2
                              /* result for -rCl */
network RC2 (terminal a, b)  | network RC2 (terminal a, b)
{                            | {
    cap 78.26f (b, a);       |     cap 78.33f (b, a);
    cap 19.35f (b, GND);     |     cap 19.28f (b, GND);
    cap 231.34f (a, GND);    |     cap 183.67f (a, GND);
}                            | }
```

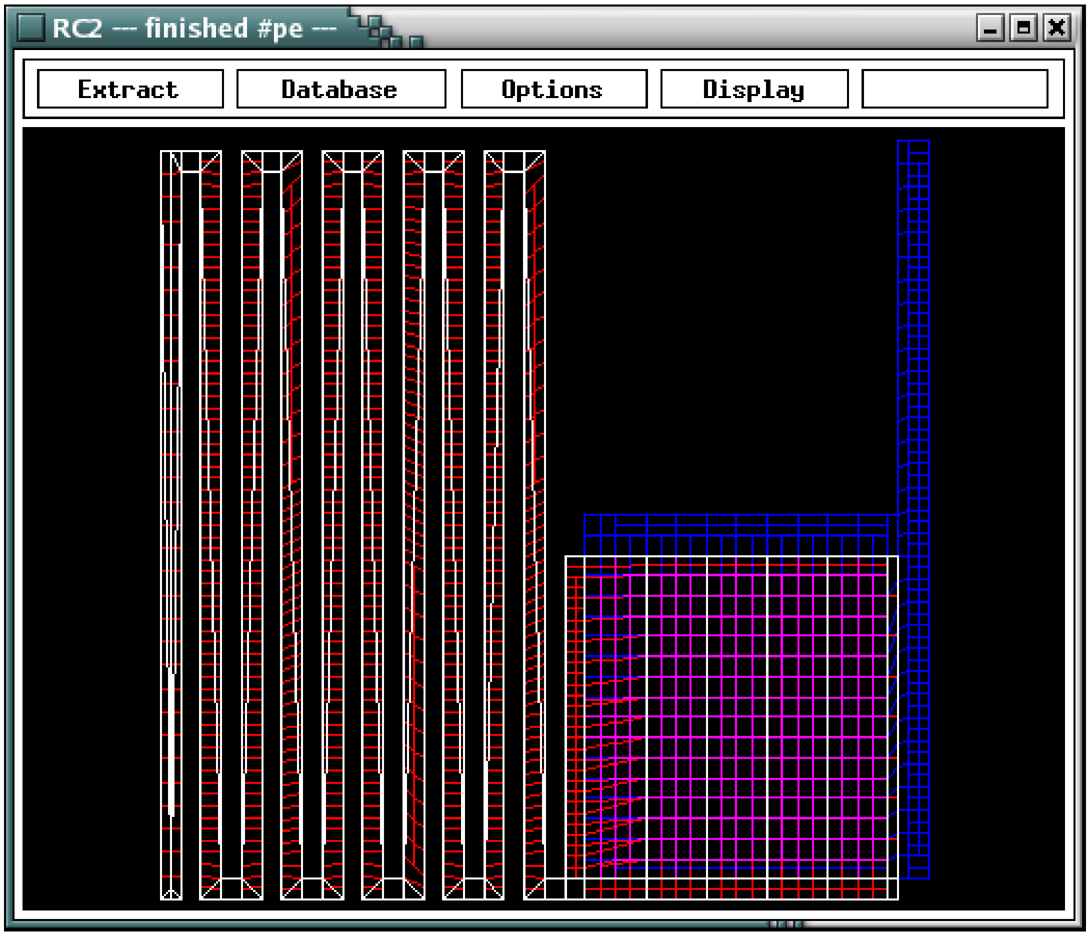
The "capacitances" section of the technology file is used for the above extractions. Thus, the above results are depending on the specified values in that section. In the above result you miss the poly-resistor. The reason for this is, that all internal nodes are eliminated. When i add an extra terminal "c" on the other side of the resistor (near the big surface cap), we get the following result (for all results, see appendix E):

```
/* result: -rCn */                  /* result: -zCn */
network RC2c (terminal c, a, b)   | network RC2c (terminal c, a, b)
{                                 | {
    cap 5.779115e-18 (b, a);      |     cap 5.488314e-18 (b, a);
    cap 78.25422f (b, c);         |     cap 78.25451f (b, c);
    cap 19.35f (b, GND);          |     cap 19.35f (b, GND);
    res 8.775016k (a, c);         |     res 9.056989k (a, c);
    cap 83.51892f (a, GND);       |     cap 83.39043f (a, GND);
    cap 147.8211f (c, GND);       |     cap 147.9496f (c, GND);
}                                 | }
```

Note that you only get the small capacitor between nodes (b, a), when you put parameter "heuristics off" (option -n). Note that terminal "c" is put on position (36, 2).

The above results are different because a different finite element mesh is used for resistance calculation options -r and -z. You can show this mesh in the following two pictures.

In the first picture you see the resistance FE mesh for option -r and the capacitance BE mesh for option -C3. The cap3D method is using the "vdimensions" and "dielectrics" data of the technology file (the data in the capacitances section is not used in this case). The result of the cap3D method can best be compared with a cap2D method where also lateral capacitances are extracted (option -l). And when all cap2D rules are defined in the technology file (with suitable values).



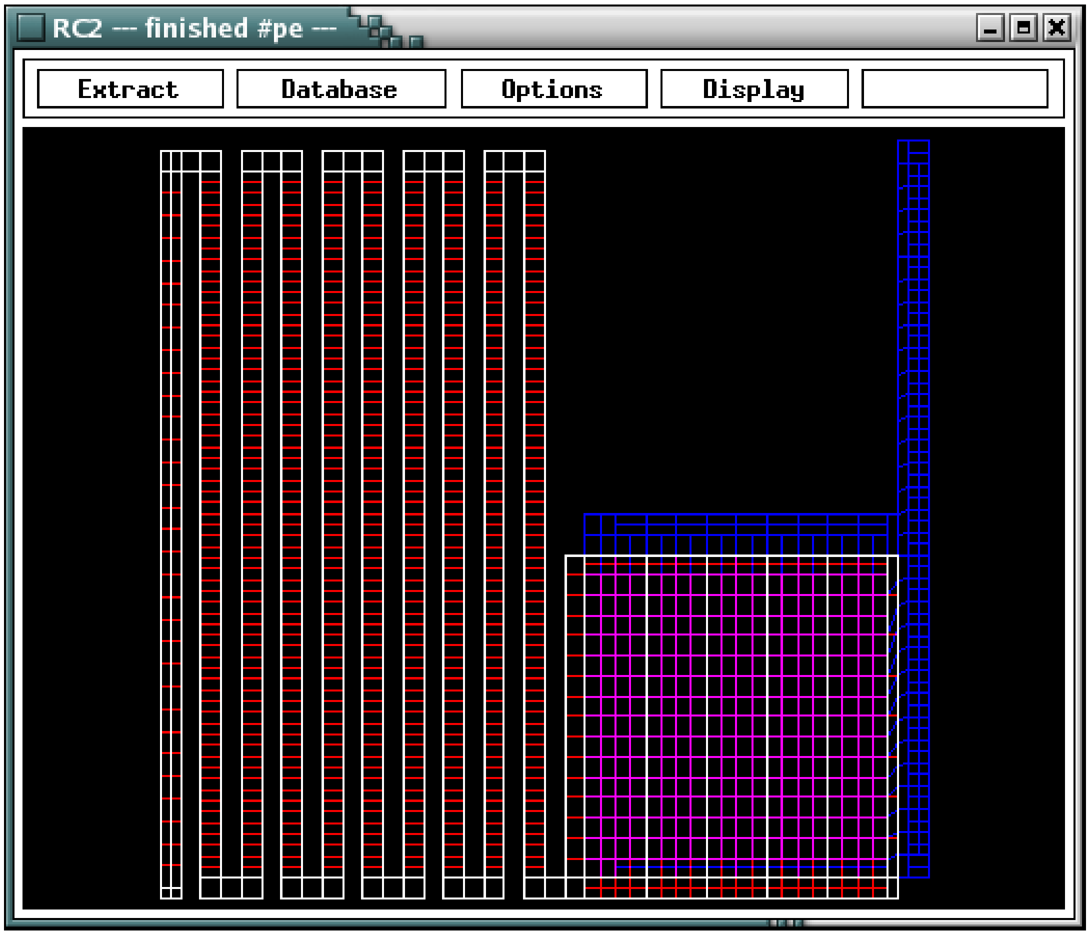
The extraction result for cap3D with -r:

```
% space3d -Espace.def.t -Pspace.def.p -r3C RC2
% xsls RC2

network RC2 (terminal a, b)
{
    cap 61.85742f (b, a);
    cap 16.10528f (b, GND);
    cap 202.8034f (a, GND);
}
```

On the left side, you see that the first tile is split into two strokes. The reason for this is, that the center point for terminal "a" is used. This gives an extra tile split (this happens also for terminal "b"). You see that a triangle mesh is generated, when there are extra node points on a tile. On the right side, there is no FE mesh to terminal "b". This is because the metal is one big conductor node. Note that the 3D cap method is using strips, the be_window is set to 6 micron. Thus you can see vertical tile splits on every 6 micron x-position from the left.

Below you see the resistance FE mesh for option -z and the capacitance BE mesh for option -C3.



The extraction result for cap3D with -z:

```
% space3d -Espace.def.t -Pspace.def.p -z3C RC2
% xsls RC2

network RC2 (terminal a, b)
{
    cap 61.85562f (b, a);
    cap 16.10015f (b, GND);
    cap 202.4087f (a, GND);
}
```

By this resistor mesh refinement method you see less triangles, because the mesh is extended horizontally and vertically when there are extra node points on the tile edges. The BE mesh for the cap3D method is also depending on these resistance tile splitting.

5. APPENDICES

APPENDIX A -- Changed Source Files

The following sources files in the "space/lump" directory are changed in 2006:

file	new	date	comment
define.h	4.14	11/01	added alternative usage of NEWGRP
elem.c	4.48	11/01	made code faster; added NEWGRP
	4.49	11/07	moved code addGndCap/SubCap to elim.c
elim.c	4.34	11/01	added new SNE method; made code faster
	4.35	11/03	put #ifndef in testRCelim, changed some code
	4.36	11/07	added code addGndCap/SubCap of elem.c
	4.37	12/13	changed sne printElimCount
extern.h	4.48	11/01	added NEWGRP; changed some function templates
init.c	4.68	09/27	added tests for SNE parameters
	4.69	11/01	added debug_caps; optimized the code; fixed error
	4.70	12/01	bug fix for size of capTransf array
lump.c	4.107	11/01	added moments2, NEWGRP;
			removed CAP_DS_TABLE, res/capSortEnable
	4.108	11/03	removed #ifndef around testRCelim; removed ASSERTs
	4.109	11/07	moved n -> moments code; added keep=1 again
node.c	4.79	09/15	added SNE code to nqEliminateGroup; added nqInsert
	4.80	11/01	added moments2, NEWGRP; optimized/changed code
	4.81	11/03	removed #ifndef around testRCelim
	4.82	11/07	repair of a mistake
	4.83	12/13	changed sne printElimCount
out.c	4.104	11/01	added moments2; improved some code
ready.c	4.35	09/15	moved SNE code to nqEliminateGroup
	4.36	11/01	added delay_cnt; improved some code
reduc.c	4.51	11/01	improved some code

APPENDIX B -- Technology File of RC2

```
# process : nmos

colors :
np      red
nm      blue

unit resistance    1     # Ohm
unit a_capacitance 1e-3  # Farad per meter^2
unit e_capacitance 1e-9  # Farad per meter
unit capacitance   1e-15 # Farad
unit vdimension    1e-6  # meter

conductors :
    resM : nm       : nm : 0.04   # the exact values for
    resP : np       : np : 25     # sheet-resistivity

capacitances :
  # capacitances to ground

    capM  : nm !np      :  nm : 0.03
    capMe : !nm -nm !np : -nm : 0.06

    capP  : np          :  np : 0.05
    capPe : !np -np     : -np : 0.06

  # coupling capacitances
    capMP  : nm np      :  nm  np : 0.07
    capMeP : !nm -nm np : -nm  np : 0.07
    capMPe : nm !np -np :  nm -np : 0.07

    CapPPl : !np !nm -np =np : -np =np : 0.07 #lateral coupling
    CapPMl : !np !nm -np =nm : -np =nm : 0.07 #lateral coupling
    CapMMl : !np !nm -nm =nm : -nm =nm : 0.07 #lateral coupling

vdimensions :
    nm_on_all : nm       : nm : 2.05 1.0
    np_on_rest: np       : np : 0.65 0.8

dielectrics :
  # Dielectric consists of 5 micron thick SiO2
  # (epsilon = 3.9) on a conducting plane.
    SiO2   3.9   0.0

#EOF
```

APPENDIX C -- Parameters File of RC2

```
# General options for space.

#heuristics              off     # default: on
min_art_degree          3       # heuristic
min_degree              4       # heuristic
min_res               100       # heuristic (ohm)
max_par_res            20       # heuristic (ratio)
min_coup_cap            0.05    # heuristic (ratio)

lat_cap_window          6.0     # micron
#low_sheet_res          0       # default: 1 ohm
#param_verbose          on      # default: off

BEGIN sne
#resolution             100     # default: 1
END moments

BEGIN moments
max                    2        # default: 2
print                  off      # default: on
#out_negc              off      # default: on
#out_negr              off      # default: on
END moments

BEGIN cap3d
be_window              6    # micron
be_shape               4    #3=triangle; 4=rectangle
be_mode                0c   #c=collocation; g=galerkin
max_be_area            4    # default for 3D extraction
END cap3d
```

APPENDIX D -- LDM File of RC2

```
:: Delft Layout Description Modified, LDM V3.2
:: Generated with the program xldm V4.08
:: at Thu Dec 12 10:46:27 1996
:: by the Delft NELSIS IC Design System Release 3
:: use cldm without -o
:: project = /u/10/10/elias/space/RC2
:: process = nmos
:: lambda = 1 micron
:: extraction of all related cells of cell: RC2
:: level 1
ms RC2
term np 0 2 0 2 a
term nm 73 76 72 75 b
box np 0 2 0 72
box np 4 6 2 72
box np 8 10 2 72
box np 12 14 2 72
box np 16 18 2 72
box np 20 22 2 72
box np 24 26 2 72
box np 28 30 2 72
box np 32 34 2 72
box np 36 38 2 72
box np 0 6 72 74
box np 8 14 72 74
box np 16 22 72 74
box np 24 30 72 74
box np 32 38 72 74
box np 28 34 0 2
box np 20 26 0 2
box np 12 18 0 2
box np 4 10 0 2
box np 40 73 2 34
box np 36 73 0 2
box nm 42 76 2 38
box nm 73 76 38 75
me
:: eof
```

APPENDIX E -- RC2c Extraction Results

```
% space3d -Espace.def.t -Pspace.def.p -rCn RC2c
```

-rCn:	-zCn:
cap 5.779115e-18 (b, a);	cap 5.488314e-18 (b, a);
cap 78.25422f (b, c);	cap 78.25451f (b, c);
cap 19.35f (b, GND);	cap 19.35f (b, GND);
res 8.775016k (a, c);	res 9.056989k (a, c);
cap 83.51892f (a, GND);	cap 83.39043f (a, GND);
cap 147.8211f (c, GND);	cap 147.9496f (c, GND);
-rC:	-zC:
cap 78.25422f (b, c);	cap 78.25451f (b, c);
cap 19.35578f (b, GND);	cap 19.35549f (b, GND);
res 8.775016k (a, c);	res 9.056989k (a, c);
cap 83.5247f (a, GND);	cap 83.39592f (a, GND);
cap 147.8211f (c, GND);	cap 147.9496f (c, GND);
-rCln:	-zCln:
cap 9.297514e-18 (b, a);	cap 8.872521e-18 (b, a);
cap 78.3207f (b, c);	cap 78.32113f (b, c);
cap 19.28f (b, GND);	cap 19.28f (b, GND);
res 8.775016k (a, c);	res 9.056989k (a, c);
cap 2.368174f (a, c);	cap 2.395123f (a, c);
cap 60.77221f (a, GND);	cap 60.68401f (a, GND);
cap 122.8978f (c, GND);	cap 122.986f (c, GND);
-rCl:	-zCl:
cap 78.3207f (b, c);	cap 78.32113f (b, c);
cap 19.2893f (b, GND);	cap 19.28887f (b, GND);
res 8.775016k (a, c);	res 9.056989k (a, c);
cap 63.14969f (a, GND);	cap 63.088f (a, GND);
cap 125.266f (c, GND);	cap 125.3811f (c, GND);
-rC3n:	-zC3n:
cap 8.232698e-18 (b, a);	cap 8.320364e-18 (b, a);
cap 61.84919f (b, c);	cap 61.8473f (b, c);
cap 16.10528f (b, GND);	cap 16.10015f (b, GND);
res 8.775016k (a, c);	res 9.056989k (a, c);
cap 1.738201f (a, c);	cap 1.998448f (a, c);
cap 69.67809f (a, GND);	cap 69.35648f (a, GND);
cap 133.1253f (c, GND);	cap 133.0522f (c, GND);
-rC3:	-zC3:
cap 61.84919f (b, c);	cap 61.8473f (b, c);
cap 16.11351f (b, GND);	cap 16.10847f (b, GND);
res 8.775016k (a, c);	res 9.056989k (a, c);
cap 71.42452f (a, GND);	cap 71.36325f (a, GND);
cap 134.8635f (c, GND);	cap 135.0507f (c, GND);