Z_SOIL on 64-bit Windows system

Z_Soil Day 2009

Krzysztof Podleś

Windows 32 bit (x86) limitations

The primary limitation of 32-bit Windows operating system is the maximum allocatable system memory (RAM).

A single process on a 32-bit Windows operating system is limited to a total of 3.25 GB (gigabytes).

$2^{32} =$ 4 294 967 296

Windows 64 bit (x64) limitations

Theoretical memory limit a 64-bit computer can address is about 16 exabytes

$$2^{64} = 16 \times 1024 \times 1024 \times 1024 \times 1024 \times 1024 \times 1024$$

18 446 744 073 709 551 616

Windows XP (x64)

Windows XP x64 is limited to :

- 128 GB of physical memory
- 8 terabytes of virtual memory per process

Windows VISTA 64 bit (x64) limitations

- All 64-bit versions of Microsoft operating systems currently impose a **16 TB limit on address space**.
- Processes created on the 64-bit editions of Windows Vista can have 8 TB in virtual memory for user processes
- 8 TB for kernel processes to create a virtual memory of 16 TB.

In terms of physical memory

- Windows Vista 64-Bit Basic supports up to **8 GB of RAM**, Windows Vista 64-Bit Home Premium **16 GB of RAM**
- Windows Vista 64-Bit Business/Enterprise/Ultimate **128 GB of RAM**.

Windows 7

Maximum physical memory

- **Home Basic 8 GB**
- **Home Premium 16 GB**
- **Professional 192 GB**
- **Enterprise 192 GB**
- **Ultimate 192 GB**

32-bit data models

short	int	long
16 –32,768 to 32,767 0 to 65,535	**32** 2,147,483,648 to 2,147,483,647 0 to 4,294,967,295	**32** 2,147,483,648 to 2,147,483,647 0 to 4,294,967,295

64-bit data models

Data model	short	int	long	long long	pointers
LLP64	**16**	**32**	**32**	**64**	**64**
LP64	**16**	**32**	**64**	**64**	**64**
ILP64	**16**	**64**	**64**	**64**	**64**
SILP64	**64**	**64**	**64**	**64**	**64**

64-bit data models in Z_Soil

- **LLP64**

Preprocesing, Menu, Postprocessig
Available total memory – as system
Maximux array in one block – 16 GB

- **LP64**

Calculation
Available total memory – as system
Maximux array in one block – 16 GB

New Compliers

- V 2009 – VC ++ 6.0
  Compaq Fortran 90 Compiler

- V 2010 Visual Studio 2008
- Intel Fortran Compiler
- VC ++ 11.0 Compiler – planned for testing

Tests

- Windows Vista Bussiness
- 8 GB of physical memory
- 100 GB of virtual memory

- **V 2009**
- **V 2010 (32-bit)**
- **V 2010 (64-bit)**

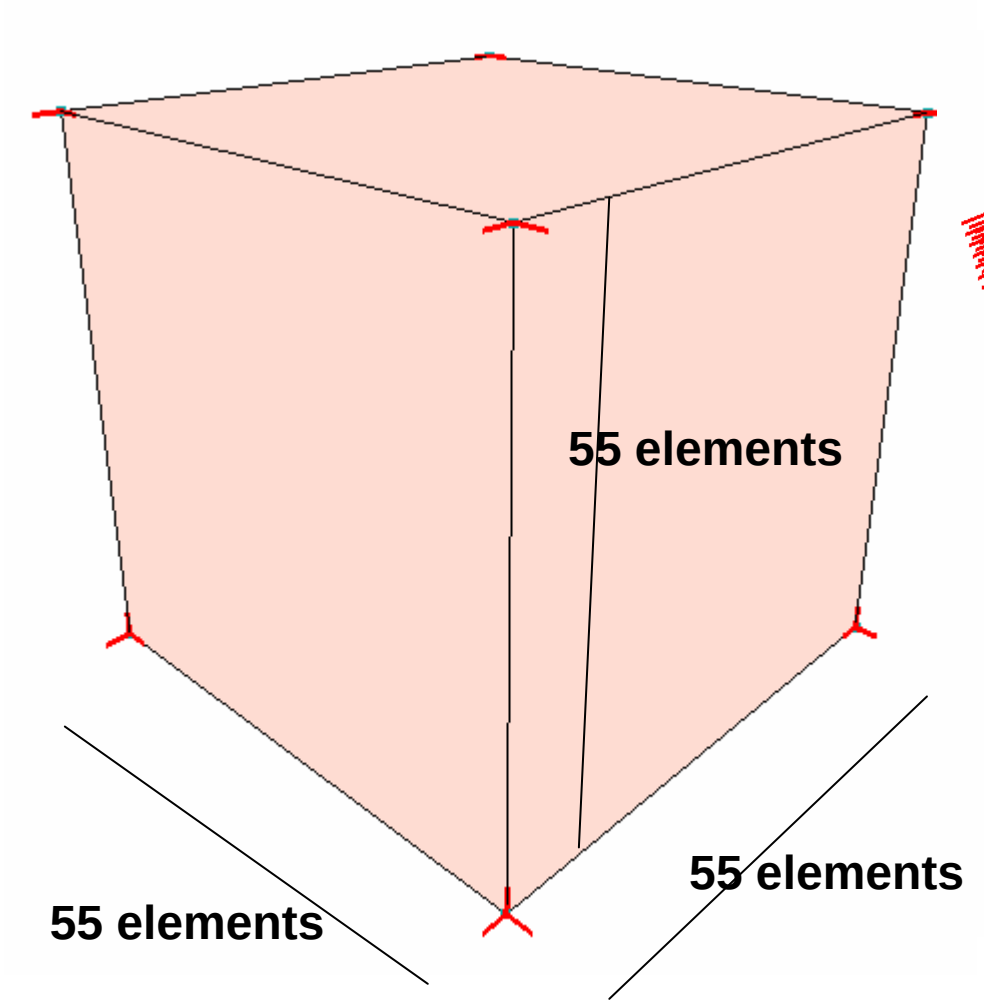
Tests

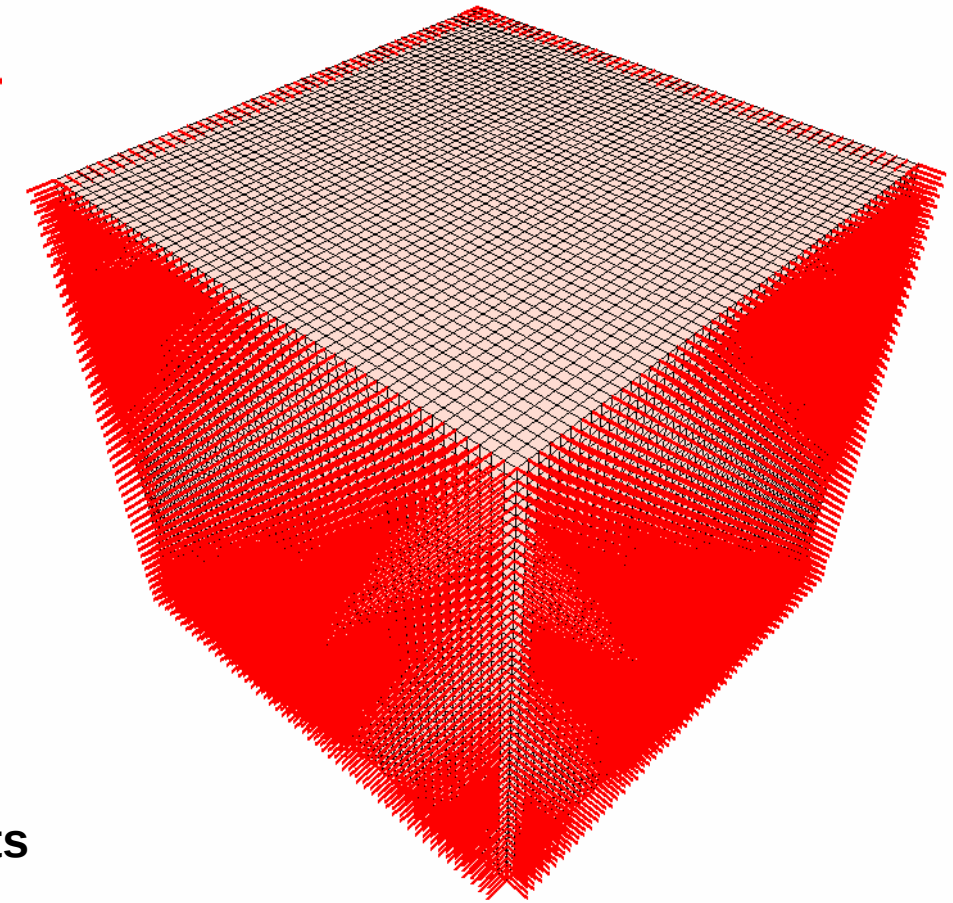
For each examples one step of time dependent driver was calulated

Total time of the solution takes into account:

- Reading *.dat
- Aggregation of LHS and RHS
- Reordering
- Factorization
- Solution
- Saving results

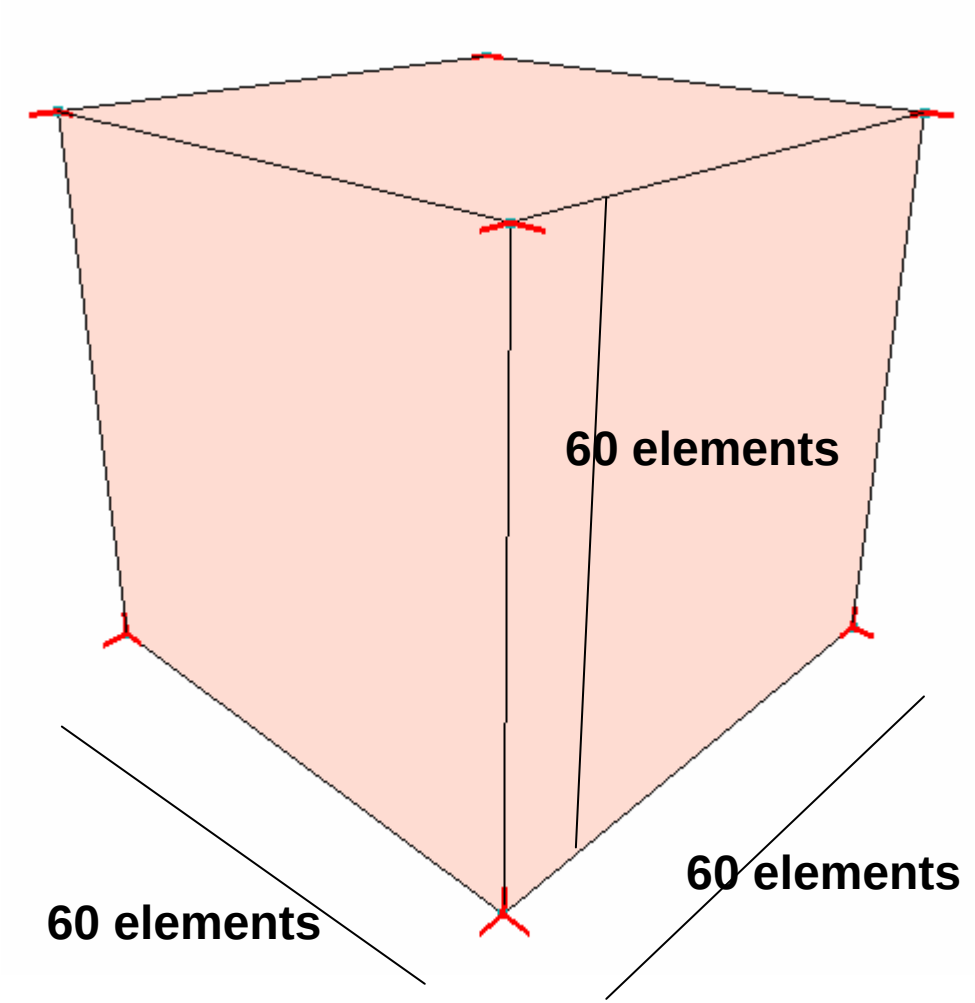
Box

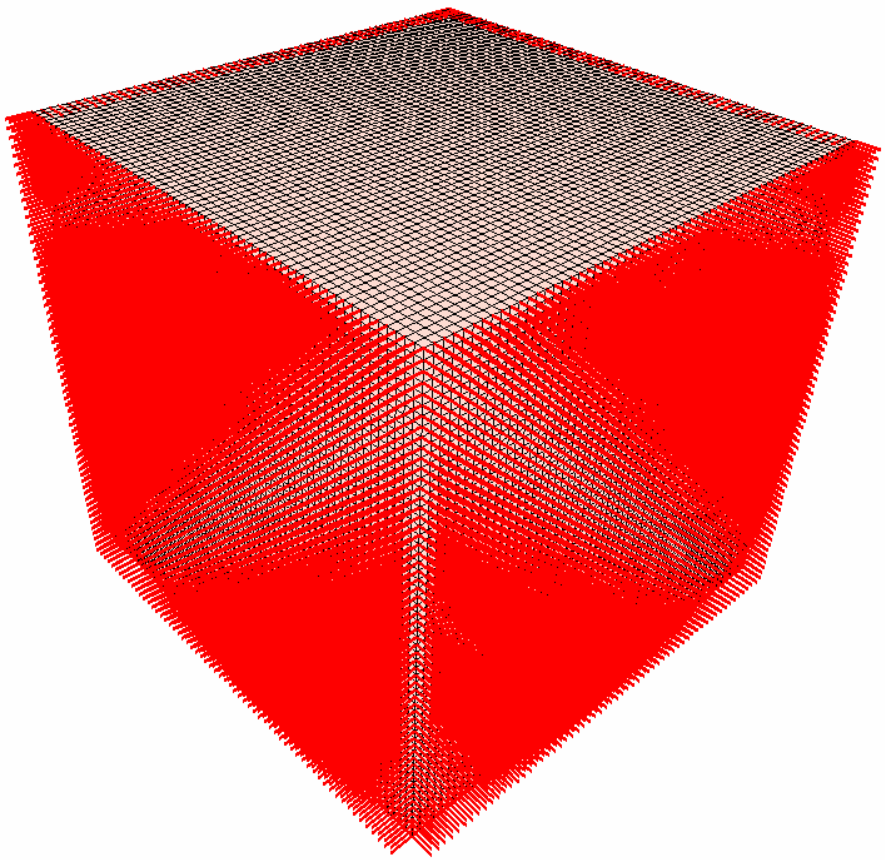




BOX_55xx55x55.inp

Box





BOX_60xx60x60.inp

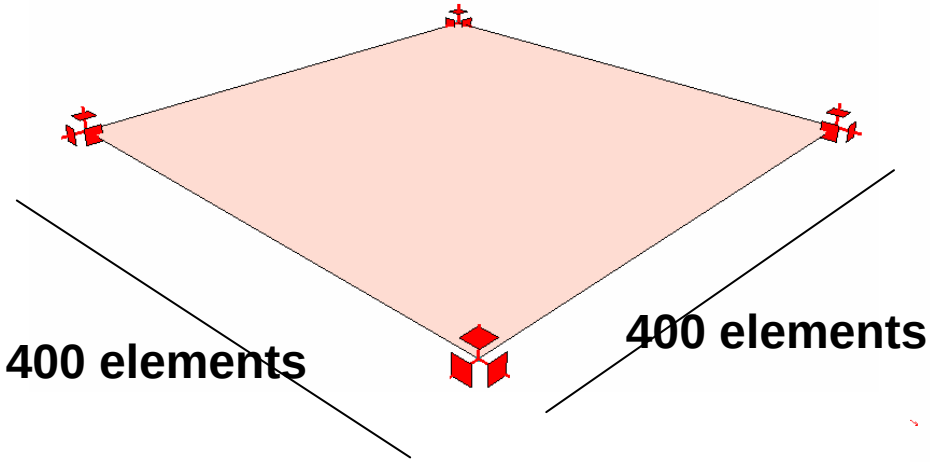
Box – results

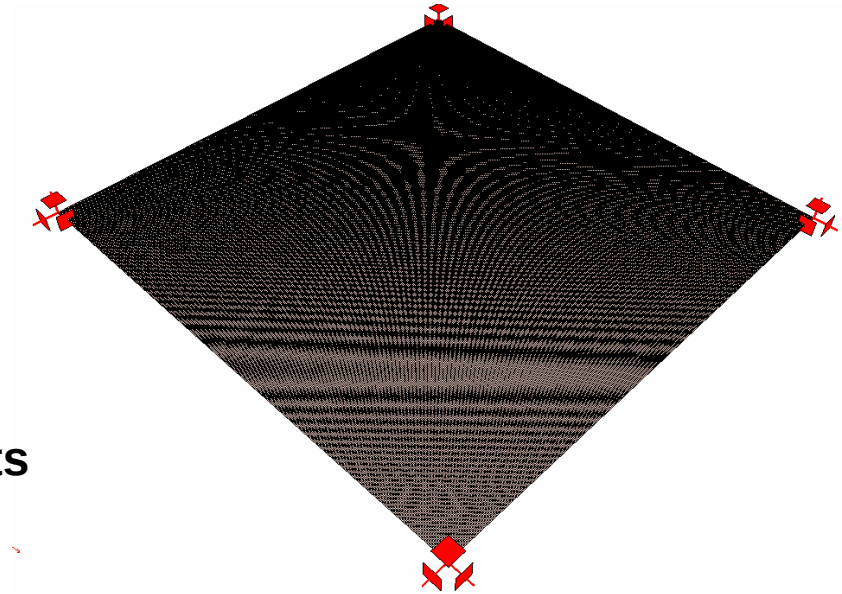
Example	Nodes	Elements	Neq Ndofs	Total time [s]		
				v2009	**v2010**	**v2010-x64**
box 55x55x55	175 616	166 375	505 120	**1498**	**1347**	**823**
box 60x60x60	226 981	216 000	655 140	**x**	**2188**	**3397**

Box nonsymmetric matrix - results

Example	Nodes	Elements	Ndofs Neq	Total time [s]		
				v2009	**v2010**	**v2010-x64**
box 55x55x55	175 616	166 375	505 120	**x**	**7154**	**3900**
box 60x60x60	226 981	216 000	655 140	**x**	**x**	**7469**

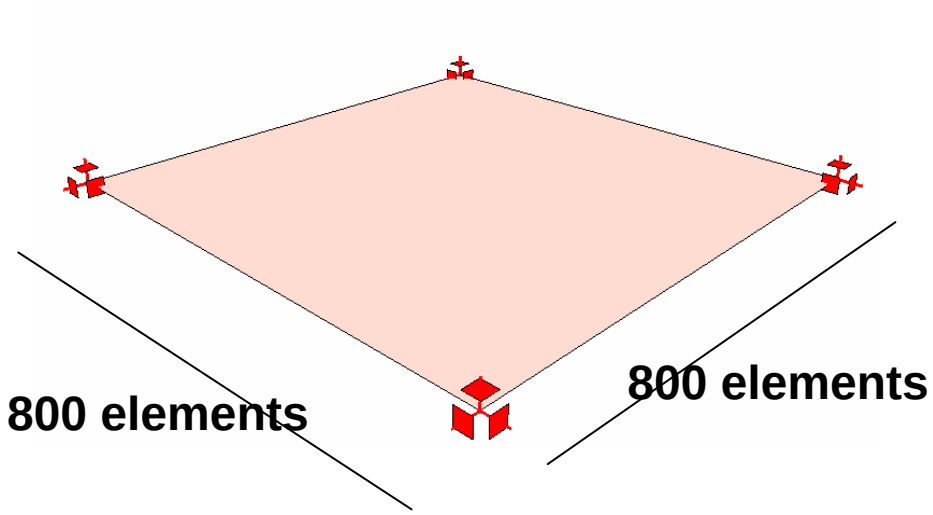
Shell





SH_400x400.inp

Shell

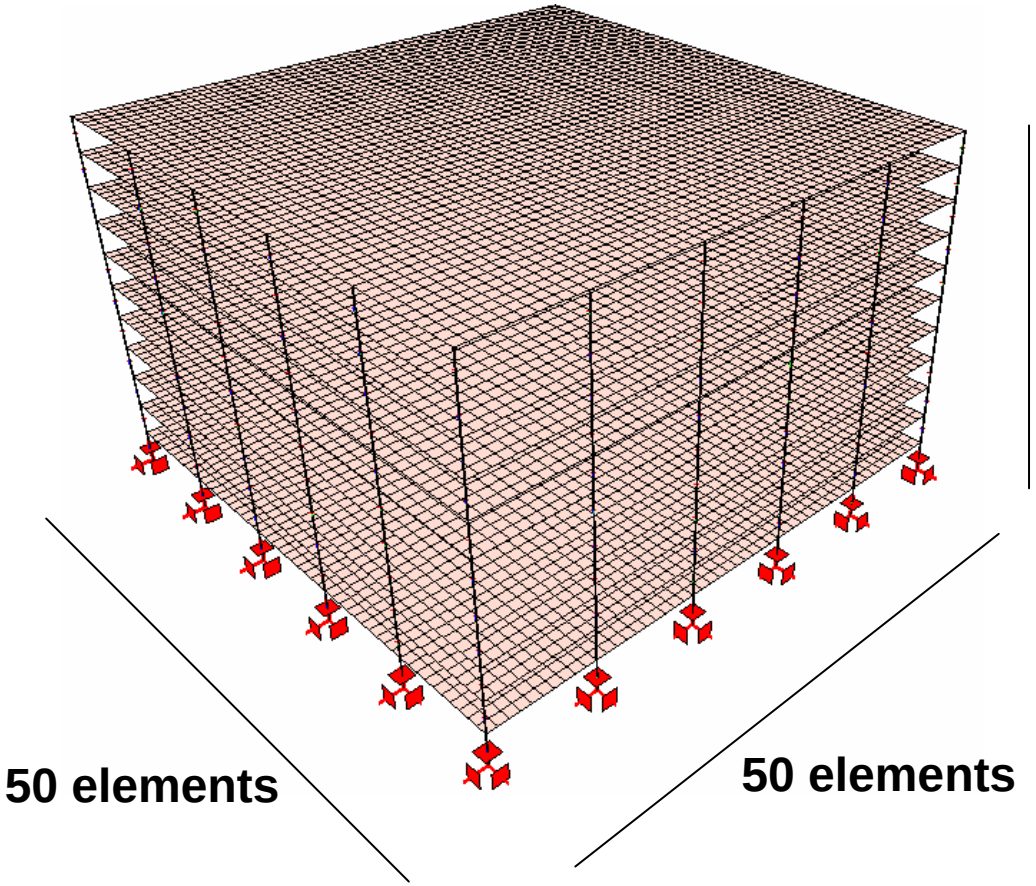


SH_800x800.inp

Shell - results

Example	Nodes	Elements	Ndofs Neq	Total time [s]		
				v2009	**v2010**	**v2010-x64**
SH_400x400	160 801	160 000	**964 782**	**470**	**351**	**174**
SH_800x800	641 601	640 000	**3 849 582**	**x**	**x**	**0/3386**

Building – shells + beams



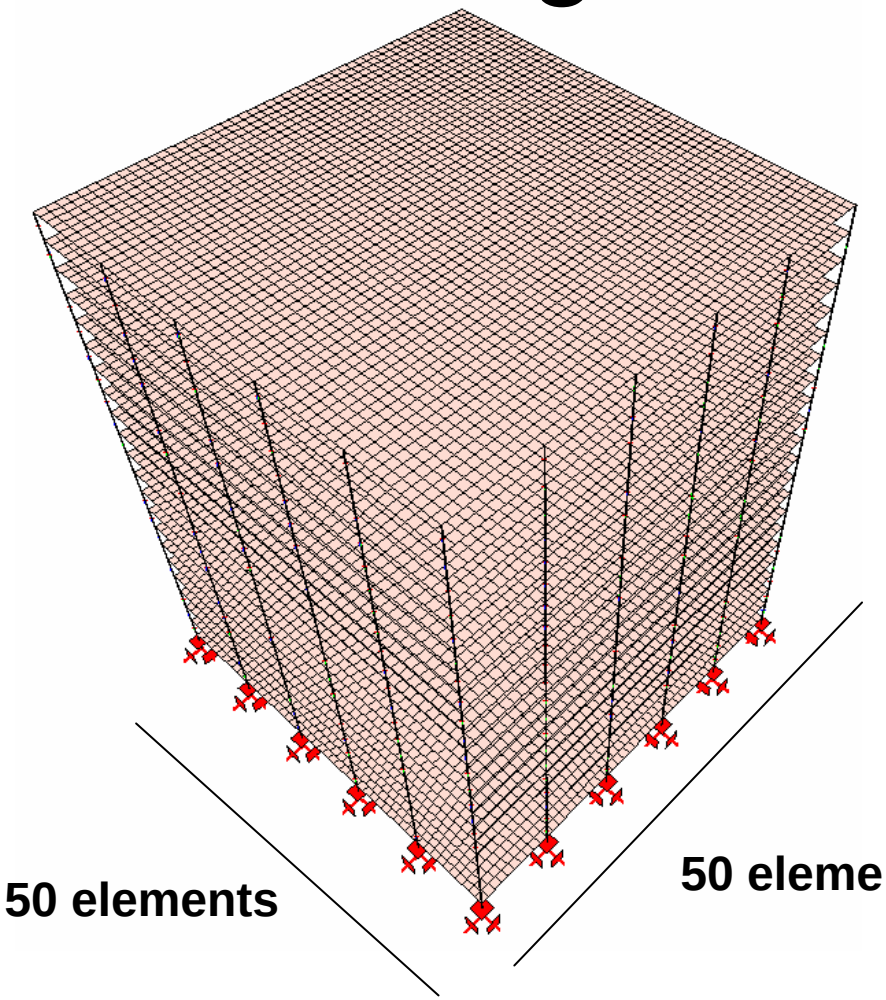
11 storey

B_50x50x11.inp

50 elements

50 elements

Building – shells + beams



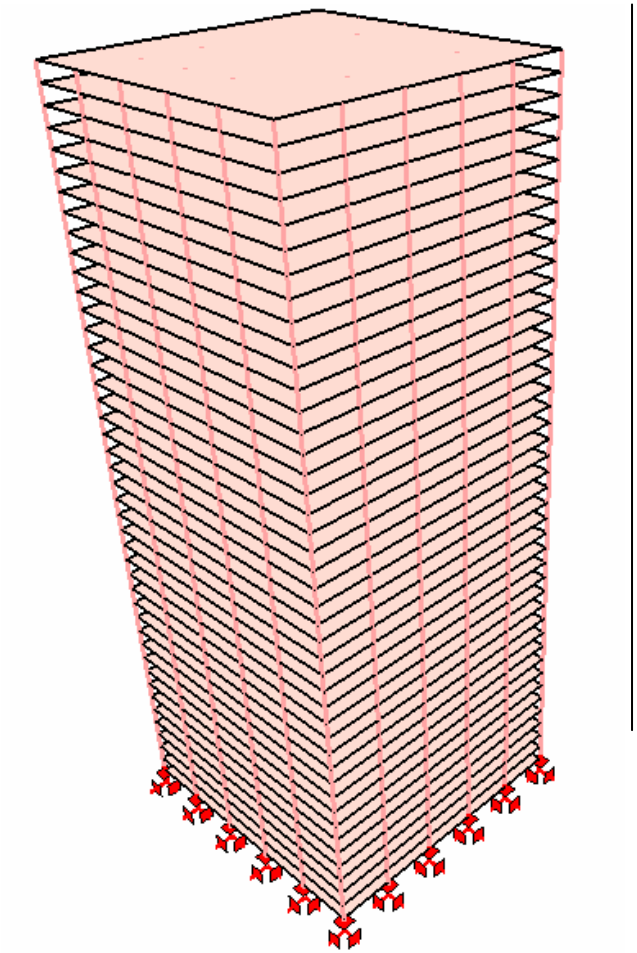
22 storey

B_50x50x22.inp

50 elements

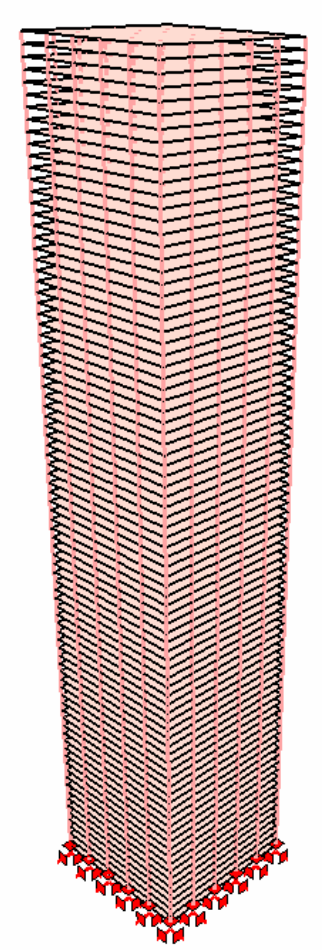
50 elements

Building – shells + beams



B_50x50x44.inp

44 storey



88 storey

B_50x50x88.inp

Building – results

Example	Nodes	Elements	Ndofs Neq	Total time [s]		
				v2009	**v2010**	**v2010-x64**
sh_50x50x11	29 439	27 896	171 666	**64**	**38**	**29**
sh_50x50x22	58 842	55 792	343 332	**127**	**81**	**58**
sh_50x50x44	111 584	117 648	686 664	**263**	**155**	**117**
sh_50x50x44	223 168	235 296	**1 373 328**	**x**	**452**	**235**

Disk or memory

- IN CORE SOLVER

  8 GB is used - no way to work with other applications

- OUT OF CORE SOLVER

  User decide how much memory can be used for application. It is slower but it is possible to work with other application simultaneously

Fighting with bottlenecks

Bottleneck is a part of the code/program, where capacity of entire system is limited

It is not optimized functions/procedure

Split B8

- 50 x 50 x 50 - 180 s reduced to 30 s
- 55 x 55 x 55 - 247 s reduced to 43 s
- 60 x 60 x 60 - 399 s reduced to 74 s

Split Q4/Shell one layer

- 400 x 400 - 6 hours reduced to 15 s
- 800 x 800 - ~week reduced to 43 s

Piles

User example with 1000 piles split by 15 and 80 000 elements

Save time in Preprocessing was reduced from 17 min to 12 s

Virtual to real

Create 100 000 elements from virtual mesh to real was reduced by factor 100