

Performance and suitability of EMME-3 Highway Assignment Modules

Matt Carlson

EMME User Group, London, UK, 2011-06-22

Contents

- **Motivation**
- **Variable Demand Model**
- **4-Stage Model**
- **Conclusions**

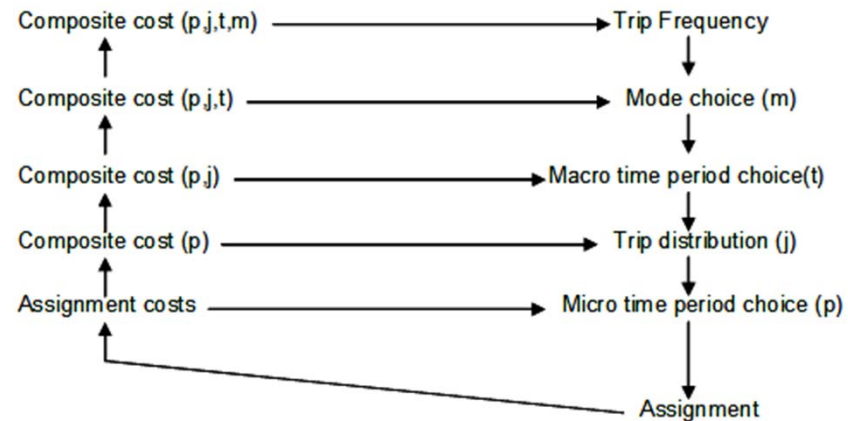
Motivation

- **New modules:**
 - 5.22 Frank & Wolfe Assignment (parallel processing)
 - 5.25 Path-based Assignment
- **We know they work in very different ways**
 - But which is the best one to use?
- **2 Recent Examples**
 - A Variable Demand Model (VDM)
 - A 4-Stage Model

Variable Demand Model (VDM)

- **An existing SATURN Model was imported to EMME**
 - Usual elements:
 - Networks
 - Matrices
 - Plus importantly:
 - Volume delay functions (Links & Turns)
- **Enable EMME to reproduce the SATURN flows and delays at a reasonable R^2 value (>0.9)**
 - Imported using Perl scripts

VDM: Use of EMME (1)



- **VDM implemented in EMME**
 - Iterative loop until cost weighted trips converge
 - See <http://www.dft.gov.uk/webtag/documents/expert/unit3.10.php>
- **Greater flexibility than DIADEM**
 - Choose which responses to use

VDM: Use of EMME (2)

- **Assignment implemented in EMME due to:**

1. Long SATURN run-times

1. Assignment
2. Simulation
3. If not converged, back to 1

2. No ‘plumbing’ between models required

- E.g. EMME and SATURN

VDM: VDM Approach

- **Using the SATURN VDFs as a shortcut:**
 1. Assign Base in EMME
 2. Assign Test in EMME
 3. Calculate Cost differences between 2 networks
 4. VDM Matrix Calculations
 5. If not converged, back to 2
 6. Optionally, re-assign in SATURN

VDM: Model Statistics

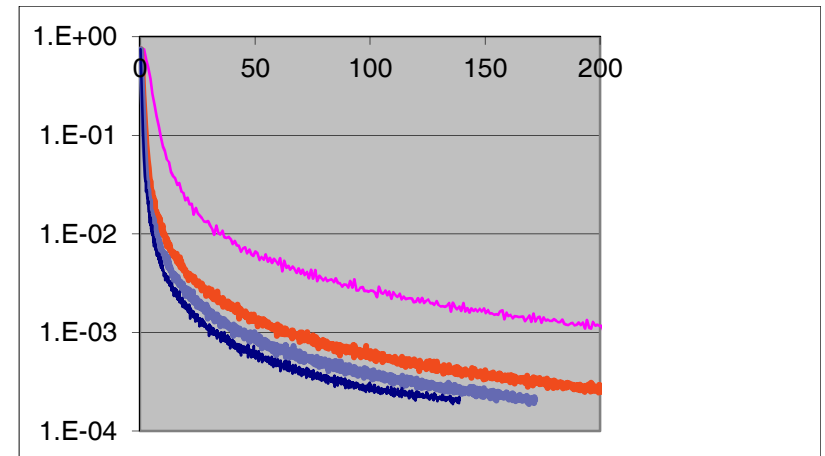
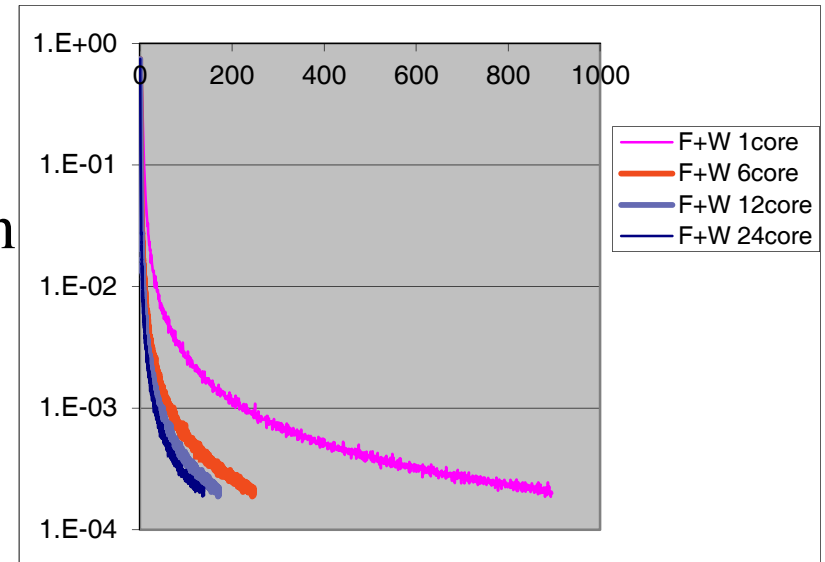
- **1400 Zones**
- **30,000 Links**
- **120,000 Turns**
- **7 User Classes**
- **Also**
 - Matrix Rounding at 10^{-1}
 - Relative Gap of 10^{-3}

VDM: Frank & Wolfe Comparisons

1000 iteration test

- **AMD Server**
 - 2x12 core 2.1GHz AMD Opteron
 - 8Gb RAM
 - Windows Server 2008 (R1)

Processors	Run Time Total (min)	Run Time per Iteration (sec)	Speed Factor
1	895	54	-
6	247	15	3.6
12	172	10	5.2
24	139	8	6.4



- **Speed-up per core tails off as cores increase**

VDM: Test Network Modifications

$$\frac{\sum_{ijctm} C(X_{ijctm}) |D(C(X_{ijctm})) - X_{ijctm}|}{\sum_{ijctm} C(X_{ijctm}) X_{ijctm}} * 100$$

Where;

X_{ijctm} is the current flow vector or matrix from the model

$C(X_{ijctm})$ is the generalised cost vector or matrix obtained by assigning that matrix

$D(C(X_{ijctm}))$ is the flow vector or matrix output by the demand model, using the costs $C(X_{ijctm})$ as input

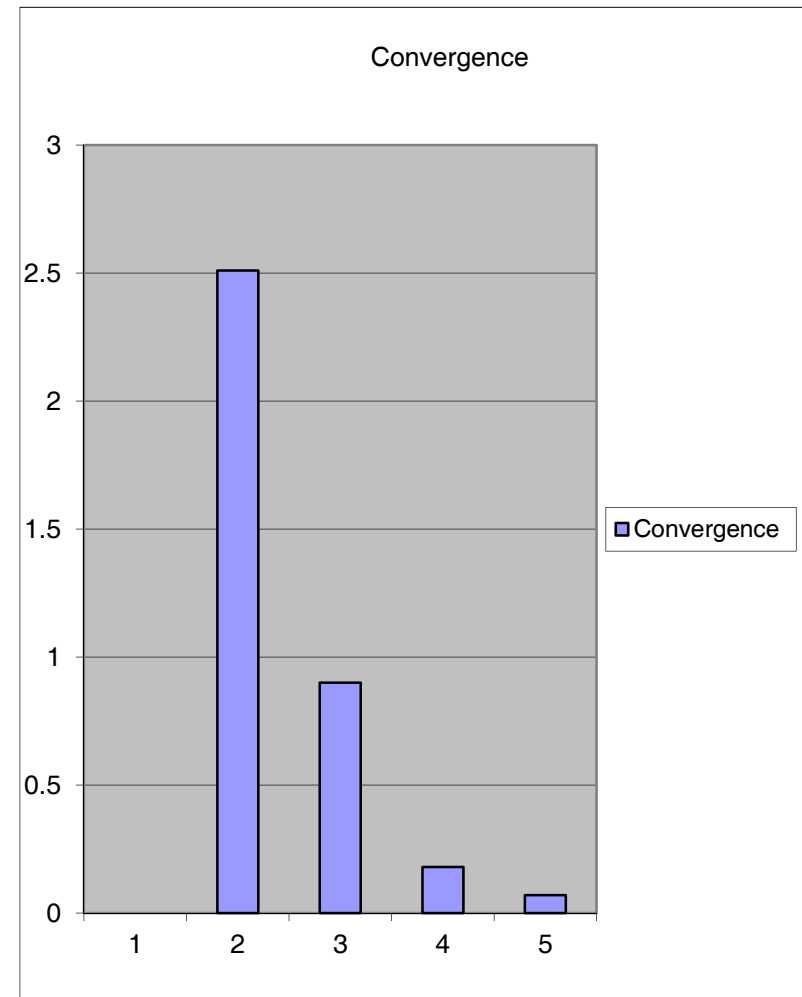
$ijctm$ represents origin i , destination j , demand segment/user class c , time period t and mode m

- **WebTAG Demand Convergence Gap set to 0.1%.**

- See <http://www.dft.gov.uk/webtag/documents/expert/unit3.10.4.php>

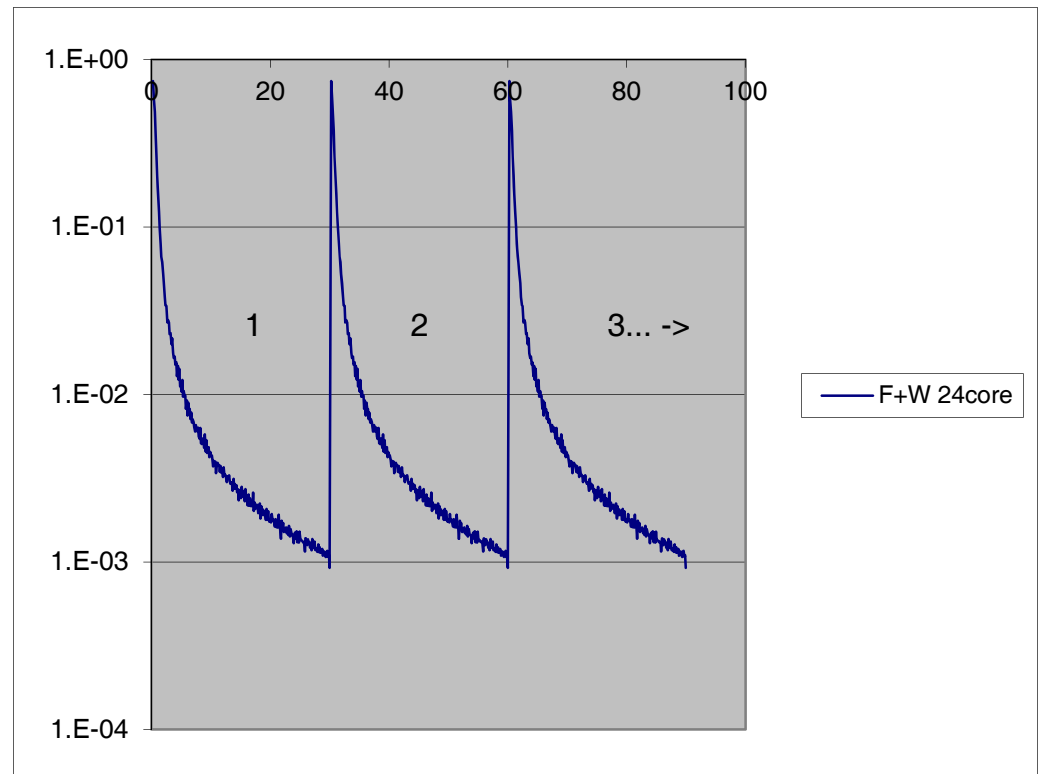
- **Achieved after 4 VDM loops, 5 test assignments, MSA on costs**

- **+ 1 base assignment = 6 assignments**



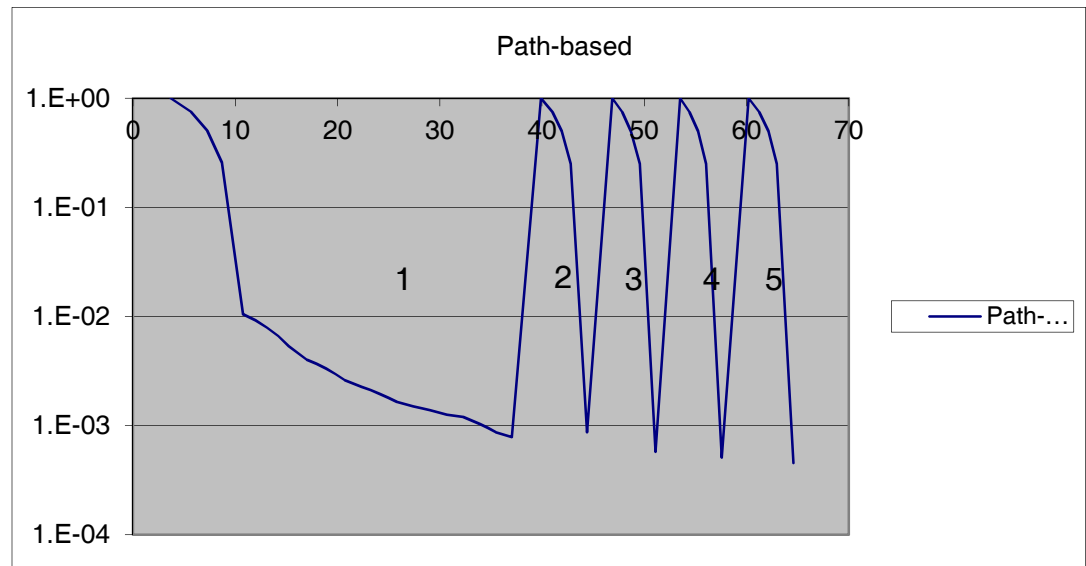
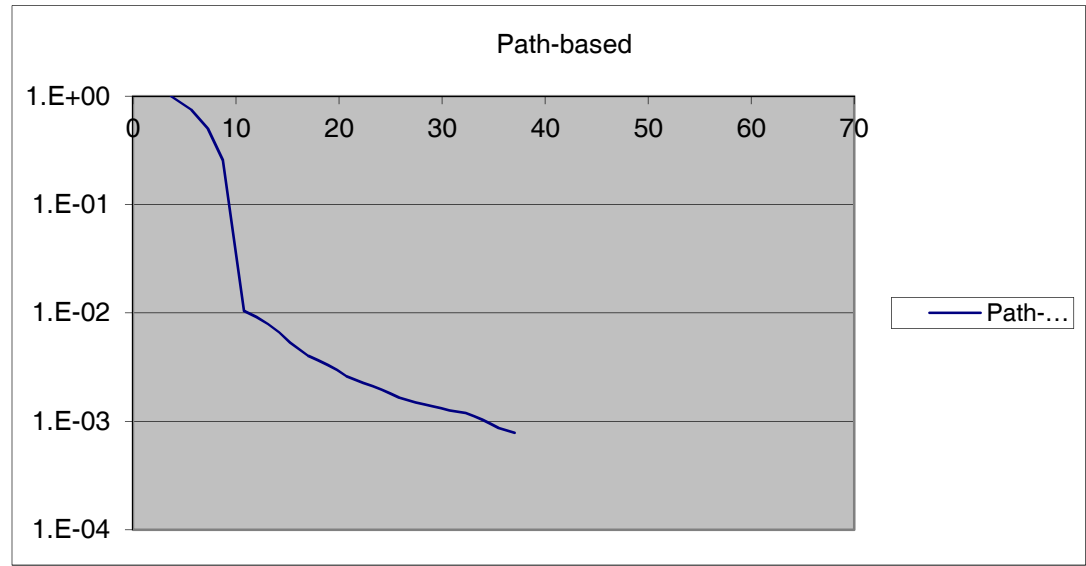
Run Time for 6 Assignments

- Each assignment of demand takes 30 minutes
- Matrix calculations not a significant time
- Total run time 3 hours



VDM: Path-Based Assignment

- **Module 5.25 on AMD**
 - 1 of 24 Processors
 - 2.1GHz AMD Opteron
 - 8Gb RAM
 - 74sec/iter (average)
 - 37 minutes for 1st Assignment
- **But:**
 - Only 65 minutes for Assignments 1 to 5.
 - Warm start on prior paths



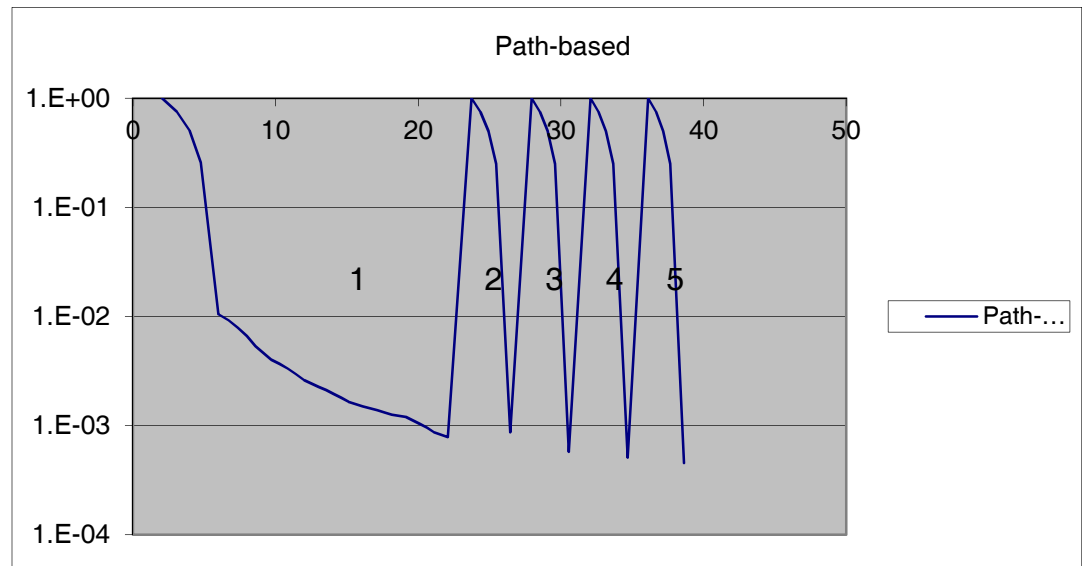
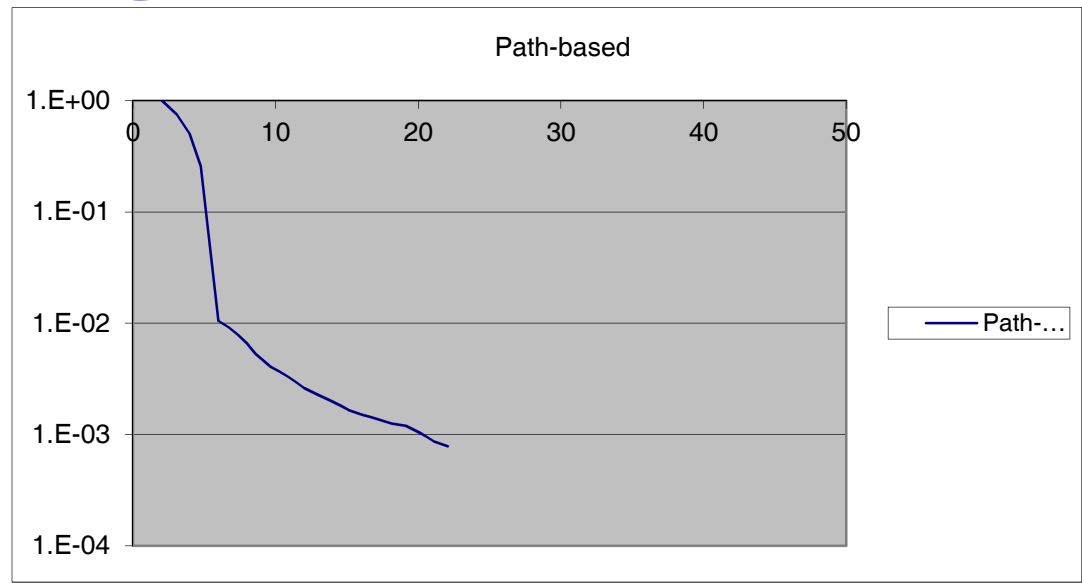
VDM: Path-Based Assignment

- **Module 5.25 on Intel**

- 1 of 4 Processors
- 3.16GHz Intel Xeon
- 8Gb RAM
- 44sec/iter (average)
- 22 minutes for 1st Assignment

- **But:**

- Only 38 minutes for Assignments 1 to 5.
- Warm start on prior paths

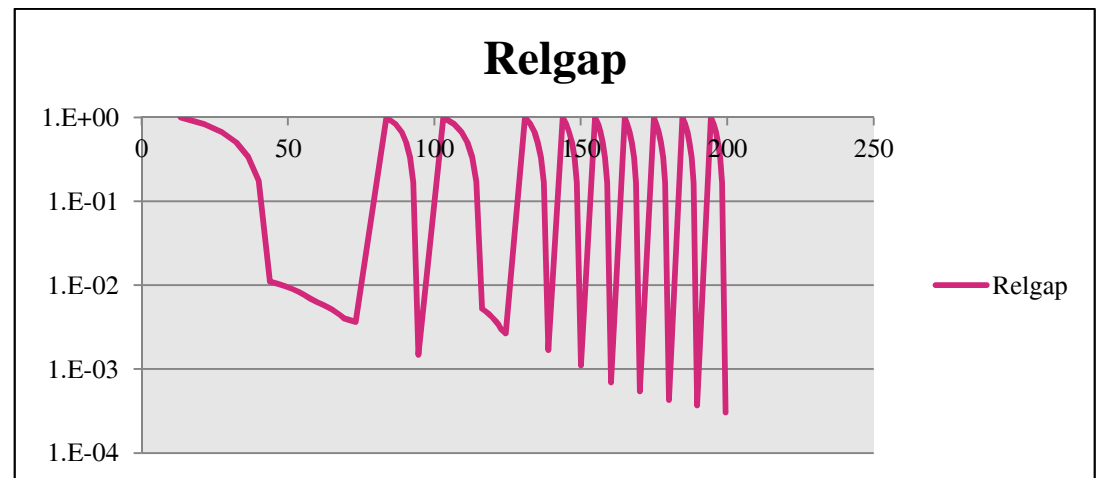
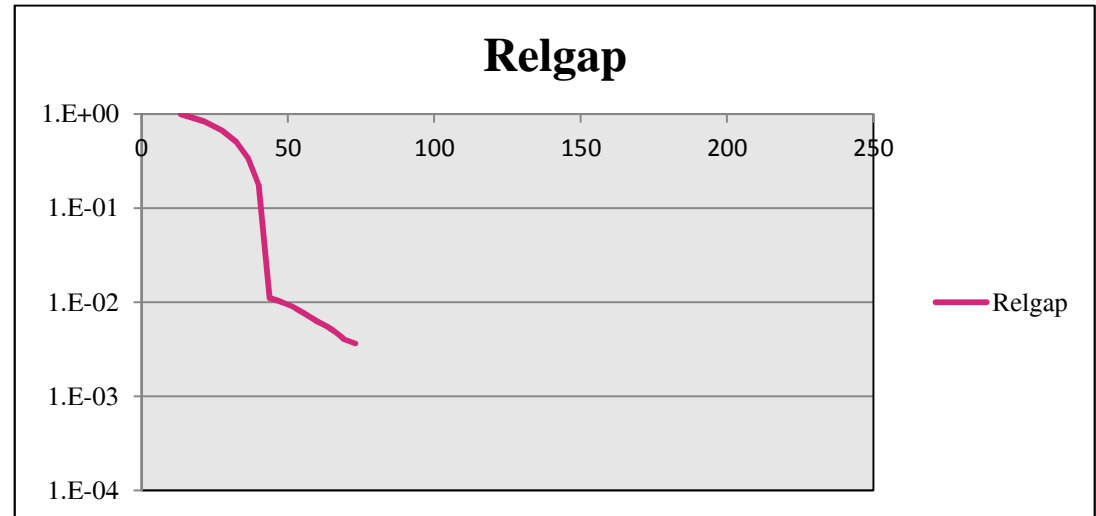


4 Stage Model Example

- **1300 zones**
- **45,000 Links**
- **20,000 Turns**
- **10 User Classes**
- **Also**
 - Matrix Rounding at 10^{-2}
 - Relative Gap of 5×10^{-3}

4 Stage Model Example

- **First Assignment
73 minutes**
- **10 Assignments in
under 200
minutes**



Conclusions

- **Path-based assignment (module 5.25) with a single core is as fast as module 5.22 with 24 cores**
- **Where demand is changing iteratively, module 5.25 is *much faster* than module 5.22**

