Tool development: automated/semi-automated

Madhu N. Belur & Narayan Rangaraj

Control & Computing, Electrical Engg
Indian Institute of Technology Bombay (IITB)

www.ee.iitb.ac.in/%7Ebelur/talks

www.ee.iitb.ac.in/%7Ebelur/railways

belur@iitb.ac.in 99 874 66 279

19th May 2021

Outline

- Simple software tools (not big software ‘packages’)
- Overview and significance of ‘home-grown’ tools, in FOSS
- Examples of semi-automated tools (decision support tool)
- ZBTT, section and network simulator by IITB

Tool conception/design

- CS/IT graduates won't come and conceive/design our tools.
- We need to find those opportunities: they (CS/IT) can code or advise.
- Tools
  - screw-driver made by manufacturing/metallurgy person
  - but conceived/thought/needed/used/designed by electrical engineers also.
  - screw-driver cum tester: is OUR domain (of usage/concept/design)
- Tools: spreadsheet, Python: good for quick and basic analysis: for us non-CS persons

Simple tools

(Tools are not software ‘packages’)

- First need to work on just a “proof of concept" preliminary tool:

Simple tools

(Tools are not software ‘packages’)

- First need to work on just a “proof of concept" preliminary tool: which would get discarded.
- Don’t plan too much: first version is going to be discarded anyway.

Simple tools

(Tools are not software ‘packages’)

- First need to work on just a “proof of concept" preliminary tool: which would get discarded.
- Don’t plan too much: first version is going to be discarded anyway.
  (At least we will start): else plan and plan and plan and then

Simple tools

(Tools are not software ‘packages’)

- First need to work on just a “proof of concept" preliminary tool: which would get discarded.
- Don’t plan too much: first version is going to be discarded anyway.
  (At least we will start): else plan and plan and plan and then we get promoted/transferred/somebody-else-takes-over
- Automatic: put in clothes in washing-machine, and they come out dried, ironed, folded, separted (??)
- Auto-pilot in planes: is not a pilot-less plane!

Simple tools

(Tools are not software ‘packages’)

- First need to work on just a “proof of concept" preliminary tool: which would get discarded.
- Don’t plan too much: first version is going to be discarded anyway.
  (At least we will start): else plan and plan and plan and then we get promoted/transferred/somebody-else-takes-over
- Automatic: put in clothes in washing-machine, and they come out dried, ironed, folded, separted (??)
- Auto-pilot in planes: is not a pilot-less plane!
  (Even driver-less cars are still distant in future:
  only fully automatic-gears are well-tested and fully licenced across the world.)
- Everything is not “automatable". Also ongoing process: slowly, steadily.
- Tool makes things partially automatic: allows repeated retrials

Simple tools

(Tools are not software ‘packages’)

- First need to work on just a “proof of concept" preliminary tool: which would get discarded.
- Don’t plan too much: first version is going to be discarded anyway.
  (At least we will start): else plan and plan and plan and then we get promoted/transferred/somebody-else-takes-over
- Automatic: put in clothes in washing-machine, and they come out dried, ironed, folded, separted (??)
- Auto-pilot in planes: is not a pilot-less plane!
  (Even driver-less cars are still distant in future:
  only fully automatic-gears are well-tested and fully licenced across the world.)
- Everything is not “automatable". Also ongoing process: slowly, steadily.
- Tool makes things partially automatic: allows repeated retrials

Simple tools

(Tools are not software 'packages')

- First need to work on just a "proof of concept" preliminary tool: which would get discarded.
- Don't plan too much: first version is going to be discarded anyway.
  (At least we will start): else plan and plan and plan and then we get promoted/transferred/somebody-else-takes-over
- Automatic: put in clothes in washing-machine, and they come out dried, ironed, folded, separted (??)
- Auto-pilot in planes: is not a pilot-less plane!
  (Even driver-less cars are still distant in future:
  only fully automatic-gears are well-tested and fully licenced across the world.)
- Everything is not "automatable". Also ongoing process: slowly, steadily.
- Tool makes things partially automatic: allows repeated retrials

Statistics (example)

- Quality of an allocation (crew allocation), comparison with the “best" possible (can use crude guidelines)
- Smaller searches rather than “brute-force" searches
- Searching for globally best often not practical for typical search sizes
- Introduce randomness at intermediate steps in the (local) search
- Many random initial starts and find best amongst final solution
- Many random initial starts and find best amongst final solution
- Home-grown suboptimal solution: better than external (closed-box) solution
- Involve (as ‘back-burner’): 3 or 4 young (by heart)/

Statistics (example)

- Quality of an allocation (crew allocation), comparison with the “best" possible (can use crude guidelines)
- Smaller searches rather than “brute-force" searches
- Searching for globally best often not practical for typical search sizes
- Introduce randomness at intermediate steps in the (local) search
- Many random initial starts and find best amongst final solution
- Many random initial starts and find best amongst final solution
- Home-grown suboptimal solution: better than external (closed-box) solution
- Involve (as ‘back-burner’): 3 or 4 young (by heart)/ dynamic/

Statistics (example)

- Quality of an allocation (crew allocation), comparison with the “best" possible (can use crude guidelines)
- Smaller searches rather than “brute-force" searches
- Searching for globally best often not practical for typical search sizes
- Introduce randomness at intermediate steps in the (local) search
- Many random initial starts and find best amongst final solution
- Many random initial starts and find best amongst final solution
- Home-grown suboptimal solution: better than external (closed-box) solution
- Involve (as ‘back-burner’): 3 or 4 young (by heart)/ dynamic/ with-criticizing ability, but not always-critic

Computation tools: example: spreadsheet/excel

- A hand-calculator (or computer's calculator) can perform complex calculations

Computation tools: example: spreadsheet/excel

- A hand-calculator (or computer's calculator) can perform complex calculations
- Insufficient for large repetitive calculations on data
- Need spreadsheet/excel's power

Computation tools: example: spreadsheet/excel

- A hand-calculator (or computer's calculator) can perform complex calculations
- Insufficient for large repetitive calculations on data
- Need spreadsheet/excel's power
  =SUM, =IF, =SUMIF, complex formula, etc.
- Can use 'VLOOKUP' to quickly merge data across sheets

Computation tools: example: spreadsheet/excel

- A hand-calculator (or computer's calculator) can perform complex calculations
- Insufficient for large repetitive calculations on data
- Need spreadsheet/excel's power
  =SUM, =IF, =SUMIF, complex formula, etc.
- Can use 'VLOOKUP' to quickly merge data across sheets
- "Advanced and mature usage of spreadsheets for large data: pivot-tables": data analysis (summary)

Computation tools: example: spreadsheet/excel

- A hand-calculator (or computer's calculator) can perform complex calculations
- Insufficient for large repetitive calculations on data
- Need spreadsheet/excel's power
  =SUM, =IF, =SUMIF, complex formula, etc.
- Can use 'VLOOKUP' to quickly merge data across sheets
- "Advanced and mature usage of spreadsheets for large data: pivot-tables": data analysis (summary)
- However, spreadsheet:

Computation tools: example: spreadsheet/excel

- A hand-calculator (or computer's calculator) can perform complex calculations
- Insufficient for large repetitive calculations on data
- Need spreadsheet/excel's power
  =SUM, =IF, =SUMIF, complex formula, etc.
- Can use 'VLOOKUP' to quickly merge data across sheets
- "Advanced and mature usage of spreadsheets for large data: pivot-tables": data analysis (summary)
- However, spreadsheet:
  - insufficient for complex allocations/constraint validation
  - has limited ability as a 'solver'
  - no 'while' loop, no jumping from one solution to another
  - not OK for automation, nor for large data
  - semi-automatic, at best

Large data problems: crew allotment

- Complex safety-based rules: Hours of Employment and Period of Rest Rules (HOER)

Large data problems: crew allotment

- Complex safety-based rules: Hours of Employment and Period of Rest Rules (HOER)
- Need to utilize crew members efficiently

Large data problems: crew allotment

- Complex safety-based rules: Hours of Employment and Period of Rest Rules (HOER)
- Need to utilize crew members efficiently
- Efficient utilization of crew: more buffer for:

Large data problems: crew allotment

- Complex safety-based rules: Hours of Employment and Period of Rest Rules (HOER)
- Need to utilize crew members efficiently
- Efficient utilization of crew: more buffer for:
  - robustness

Large data problems: crew allotment

- Complex safety-based rules: Hours of Employment and Period of Rest Rules (HOER)
- Need to utilize crew members efficiently
- Efficient utilization of crew: more buffer for:
  - robustness
  - can allow liberal leave policies

Large data problems: crew allotment

- Complex safety-based rules: Hours of Employment and Period of Rest Rules (HOER)
- Need to utilize crew members efficiently
- Efficient utilization of crew: more buffer for:
  - robustness
  - can allow liberal leave policies
- WR suburban services: $\sim$1400 services:

Large data problems: crew allotment

- Complex safety-based rules: Hours of Employment and Period of Rest Rules (HOER)
- Need to utilize crew members efficiently
- Efficient utilization of crew: more buffer for:
  - robustness
  - can allow liberal leave policies
- WR suburban services: $\sim$1400 services: to manage in $\sim$380 crew members

Large data problems: crew allotment

- Complex safety-based rules: Hours of Employment and Period of Rest Rules (HOER)
- Need to utilize crew members efficiently
- Efficient utilization of crew: more buffer for:
  - robustness
  - can allow liberal leave policies
- WR suburban services: $\sim$1400 services: to manage in $\sim$380 crew members
- Tool helps to:
  - tweak/modify and re-run: takes a minute for each program-run
  - add/delete services and re-run program
  - redecide/change lobby locations and check

Large data problems: crew allotment

- Complex safety-based rules: Hours of Employment and Period of Rest Rules (HOER)
- Need to utilize crew members efficiently
- Efficient utilization of crew: more buffer for:
  - robustness
  - can allow liberal leave policies
- WR suburban services: $\sim$1400 services: to manage in $\sim$380 crew members
- Tool helps to:
  - tweak/modify and re-run: takes a minute for each program-run
  - add/delete services and re-run program
  - redecide/change lobby locations and check
  - optimize required number of lobby locations

Large data problems: metro/suburban timetabling

- Input: services/frequencies
- Constraints: headway, turn-around, platform occupation

Large data problems: metro/suburban timetabling

- Input: services/frequencies
- Constraints: headway, turn-around, platform occupation & vacating

Large data problems: metro/suburban timetabling

- Input: services/frequencies
- Constraints: headway, turn-around, platform occupation & vacating
- Output: rake-cycles and the timetable

Large data problems: metro/suburban timetabling

- Input: services/frequencies
- Constraints: headway, turn-around, platform occupation & vacating
- Output: rake-cycles and the timetable
- Number of rakes: need to optimize

Large data problems: metro/suburban timetabling

- Input: services/frequencies
- Constraints: headway, turn-around, platform occupation & vacating
- Output: rake-cycles and the timetable
- Number of rakes: need to optimize
- Constructive timetables: rakes/services are incrementally introduced

Large data problems: metro/suburban timetabling

- Input: services/frequencies
- Constraints: headway, turn-around, platform occupation & vacating
- Output: rake-cycles and the timetable
- Number of rakes: need to optimize
- Constructive timetables: rakes/services are incrementally introduced
- Can handle rake-cycles constructively:

Large data problems: metro/suburban timetabling

- Input: services/frequencies
- Constraints: headway, turn-around, platform occupation & vacating
- Output: rake-cycles and the timetable
- Number of rakes: need to optimize
- Constructive timetables: rakes/services are incrementally introduced
- Can handle rake-cycles constructively: still difficult task

Constrained programming based solvers:

Large data problems: metro/suburban timetabling

- Input: services/frequencies
- Constraints: headway, turn-around, platform occupation & vacating
- Output: rake-cycles and the timetable
- Number of rakes: need to optimize
- Constructive timetables: rakes/services are incrementally introduced
- Can handle rake-cycles constructively: still difficult task

Constrained programming based solvers:

- Gets a good feasible solution

Large data problems: metro/suburban timetabling

- Input: services/frequencies
- Constraints: headway, turn-around, platform occupation & vacating
- Output: rake-cycles and the timetable
- Number of rakes: need to optimize
- Constructive timetables: rakes/services are incrementally introduced
- Can handle rake-cycles constructively: still difficult task

Constrained programming based solvers:

- Gets a good feasible solution
- Helps to have spreadsheet based validators/checkers

(Though solvers have ensured satisfaction of constraints, one can validate by introducing 'test-flaws')

Tools: constructive/solvers/simulation/semi-automate

In my understanding

Tools: constructive/solvers/simulation/semi-automate

In my understanding

- Constructive allocation tools: ensure constraints are satisfied:

Tools: constructive/solvers/simulation/semi-automate

In my understanding

- Constructive allocation tools: ensure constraints are satisfied:
- Allocation based on 'solvers': satisfy all constraints (feasible solutions),

Tools: constructive/solvers/simulation/semi-automate

In my understanding

- Constructive allocation tools: ensure constraints are satisfied:
- Allocation based on 'solvers': satisfy all constraints (feasible solutions),
  search for 'best' feasible solution

Tools: constructive/solvers/simulation/semi-automate

In my understanding

- Constructive allocation tools: ensure constraints are satisfied:
- Allocation based on 'solvers': satisfy all constraints (feasible solutions),
  search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet

Tools: constructive/solvers/simulation/semi-automate

In my understanding

- Constructive allocation tools: ensure constraints are satisfied:
- Allocation based on 'solvers': satisfy all constraints (feasible solutions),
  search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet
- Simulation: for complex scenarios/rules/constraints
- Analysis: tool extracts statistics from simulation or other data
- Visualization tools: to ratify data

Tools: constructive/solvers/simulation/semi-automate

In my understanding

- Constructive allocation tools: ensure constraints are satisfied:
- Allocation based on 'solvers': satisfy all constraints (feasible solutions),
  search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet
- Simulation: for complex scenarios/rules/constraints
- Analysis: tool extracts statistics from simulation or other data
- Visualization tools: to ratify data
  and also view analysis-based statistics,

Tools: constructive/solvers/simulation/semi-automate

In my understanding

- Constructive allocation tools: ensure constraints are satisfied:
- Allocation based on 'solvers': satisfy all constraints (feasible solutions),
  search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet
- Simulation: for complex scenarios/rules/constraints
- Analysis: tool extracts statistics from simulation or other data
- Visualization tools: to ratify data
  and also view analysis-based statistics, aid further tool-usage

Tools: constructive/solvers/simulation/semi-automate

In my understanding

- Constructive allocation tools: ensure constraints are satisfied:
- Allocation based on 'solvers': satisfy all constraints (feasible solutions),
  search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet
- Simulation: for complex scenarios/rules/constraints
- Analysis: tool extracts statistics from simulation or other data
- Visualization tools: to ratify data
  and also view analysis-based statistics, aid further tool-usage
- Semi-automate: provide portal/interface to simplify complex problems

Tools: constructive/solvers/simulation/semi-automate

In my understanding

- Constructive allocation tools: ensure constraints are satisfied:
- Allocation based on 'solvers': satisfy all constraints (feasible solutions),
  search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet
- Simulation: for complex scenarios/rules/constraints
- Analysis: tool extracts statistics from simulation or other data
- Visualization tools: to ratify data
  and also view analysis-based statistics, aid further tool-usage
- Semi-automate: provide portal/interface to simplify complex problems
  portal takes inputs manually,

Tools: constructive/solvers/simulation/semi-automate

In my understanding

- Constructive allocation tools: ensure constraints are satisfied:
- Allocation based on 'solvers': satisfy all constraints (feasible solutions),
  search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet
- Simulation: for complex scenarios/rules/constraints
- Analysis: tool extracts statistics from simulation or other data
- Visualization tools: to ratify data
  and also view analysis-based statistics, aid further tool-usage
- Semi-automate: provide portal/interface to simplify complex problems
  portal takes inputs manually,
  after showing statistics, awaits further inputs

Machine Learning/Artificial-intelligence

Data available now for ‘pattern-searching’: can use ML and AI

Machine Learning/Artificial-intelligence

Data available now for 'pattern-searching': can use ML and AI

Only humans (natural intelligence) can conceive/design these AI tools

- Supervised learning: needs training data, validation data, and then "deploy" to find

Machine Learning/Artificial-intelligence

Data available now for 'pattern-searching': can use ML and AI

Only humans (natural intelligence) can conceive/design these AI tools

- Supervised learning: needs training data, validation data, and then "deploy" to find more/better patterns, adapt and predict
  - Used to extrapolate/predict/fit new data into pre-calculated models
  - Find outliers (in timetabling?) too fast/too slow?

Machine Learning/Artificial-intelligence

Data available now for 'pattern-searching': can use ML and AI

Only humans (natural intelligence) can conceive/design these AI tools

- Supervised learning: needs training data, validation data, and then "deploy" to find more/better patterns, adapt and predict
  - Used to extrapolate/predict/fit new data into pre-calculated models
  - Find outliers (in timetabling?) too fast/too slow?
- Unsupervised learning: no initial training: run heuristics to get good solution

Machine Learning/Artificial-intelligence

Data available now for 'pattern-searching': can use ML and AI

Only humans (natural intelligence) can conceive/design these AI tools

- Supervised learning: needs training data, validation data, and then "deploy" to find more/better patterns, adapt and predict
  - Used to extrapolate/predict/fit new data into pre-calculated models
  - Find outliers (in timetabling?) too fast/too slow?
- Unsupervised learning: no initial training: run heuristics to get good solution
  - grouping of non-daily trains into same/almost-same path

Machine Learning/Artificial-intelligence

Data available now for 'pattern-searching': can use ML and AI

Only humans (natural intelligence) can conceive/design these AI tools

- Supervised learning: needs training data, validation data, and then "deploy" to find more/better patterns, adapt and predict
  - Used to extrapolate/predict/fit new data into pre-calculated models
  - Find outliers (in timetabling?) too fast/too slow?
- Unsupervised learning: no initial training: run heuristics to get good solution
  - grouping of non-daily trains into same/almost-same path
- Reinforcement learning: specify a 'reward function' on each solution, and iterate/jump from a solution to another

Machine Learning/Artificial-intelligence

Data available now for 'pattern-searching': can use ML and AI

Only humans (natural intelligence) can conceive/design these AI tools

- Supervised learning: needs training data, validation data, and then "deploy" to find more/better patterns, adapt and predict
  - Used to extrapolate/predict/fit new data into pre-calculated models
  - Find outliers (in timetabling?) too fast/too slow?
- Unsupervised learning: no initial training: run heuristics to get good solution
  - grouping of non-daily trains into same/almost-same path
- Reinforcement learning: specify a 'reward function' on each solution, and iterate/jump from a solution to another
  often computationally easy to jump (rather than exhaustively search)

Machine Learning/Artificial-intelligence

Data available now for 'pattern-searching': can use ML and AI
Only humans (natural intelligence) can conceive/design these AI tools

- Supervised learning: needs training data, validation data, and then "deploy" to find more/better patterns, adapt and predict
  - Used to extrapolate/predict/fit new data into pre-calculated models
  - Find outliers (in timetabling?) too fast/too slow?
- Unsupervised learning: no initial training: run heuristics to get good solution
  - grouping of non-daily trains into same/almost-same path
- Reinforcement learning: specify a 'reward function' on each solution, and iterate/jump from a solution to another
  often computationally easy to jump (rather than exhaustively search)
  - optimize crew allotment
  - choose start-timings to get 'better and better' grouping at congested section

ZBTT (for GQD)

- Start-timings decided by ZR, and path-charting by IITB tool
- Maintenance block: either taken from suggestion or extracted from a timetable
- Freight windows extracted from a simulation-coaching train timetable and then freight paths charted (w.r.t. a given ratio of freight characteristics)
- Shared resources across multiple routes (with an overlap of routes)

Challenges: volume of simulation

- For one day of daily paths, need to simulate FIVE full days.

Challenges: volume of simulation

- For one day of daily paths, need to simulate FIVE full days.
- On any day (say 19th-May), trains starting on 19th share resources with
  trains that started on 17th, 18th,

Challenges: volume of simulation

- For one day of daily paths, need to simulate FIVE full days.
- On any day (say 19th-May), trains starting on 19th share resources with
  trains that started on 17th, 18th, and those starting on 20th and 21st May.
- Thus: for about 1800 daily paths of any day (of GQD), need to simulate about 9000 daily paths: lowest priority trains of 19th May will find a path after tabulating 5-days of daily paths of higher priority trains.
- Trains are simulated one by one and

Challenges: volume of simulation

- For one day of daily paths, need to simulate FIVE full days.
- On any day (say 19th-May), trains starting on 19th share resources with
  trains that started on 17th, 18th, and those starting on 20th and 21st May.
- Thus: for about 1800 daily paths of any day (of GQD), need to simulate about 9000 daily paths: lowest priority trains of 19th May will find a path after tabulating 5-days of daily paths of higher priority trains.
- Trains are simulated one by one and need to discard intermediate path-searches and move to the next
  (Possibly bad start-time for one or two trains).

Section simulation challenges

Section simulation tool for capacity estimation and timetabling

- One long section, with just one zero-milepost
- Trains go either start to end, or start/stop somewhere midway
- Trains need to decelerate/accelerate depending on
  - Multi-aspect signalling
  - Train-max-speed, accln/decln abilities
  - Section-speed, intermediate block signalling,
  - Automatic/absolute block working
  - Running lines in stations: max-speeds of loops (possibly lower than main-line)
  - Permanent Speed Restriction (PSR)
- Challenges: priority based overtaking rule

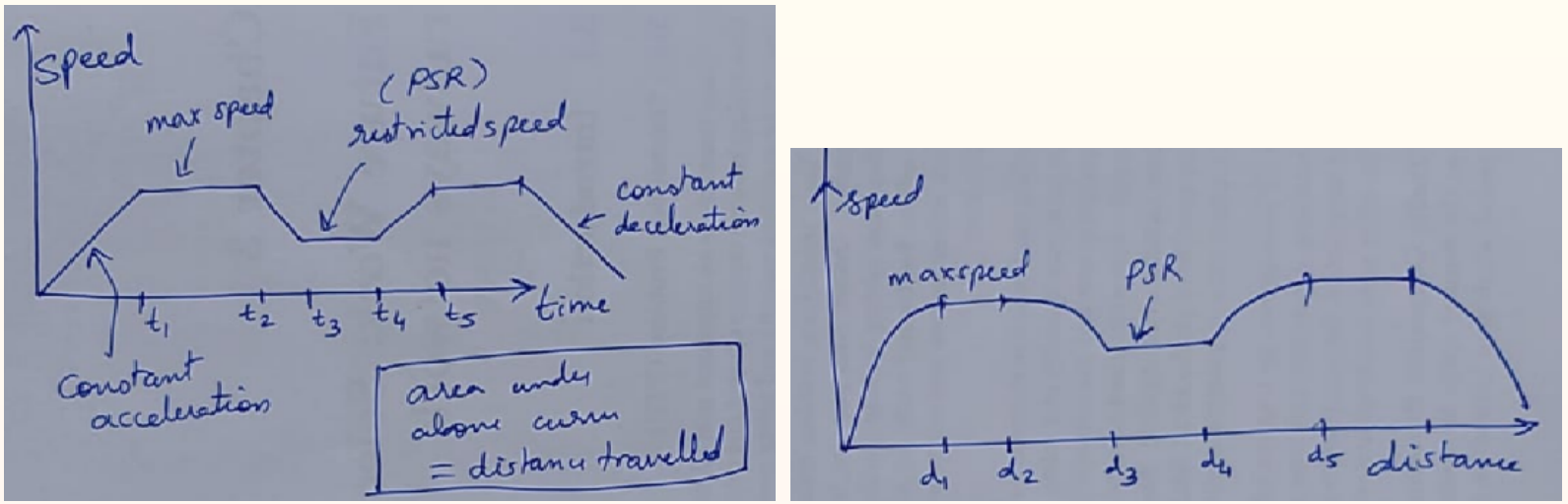


Figure: Speed vs time (more intuitive), and vs distance (for timetabling)

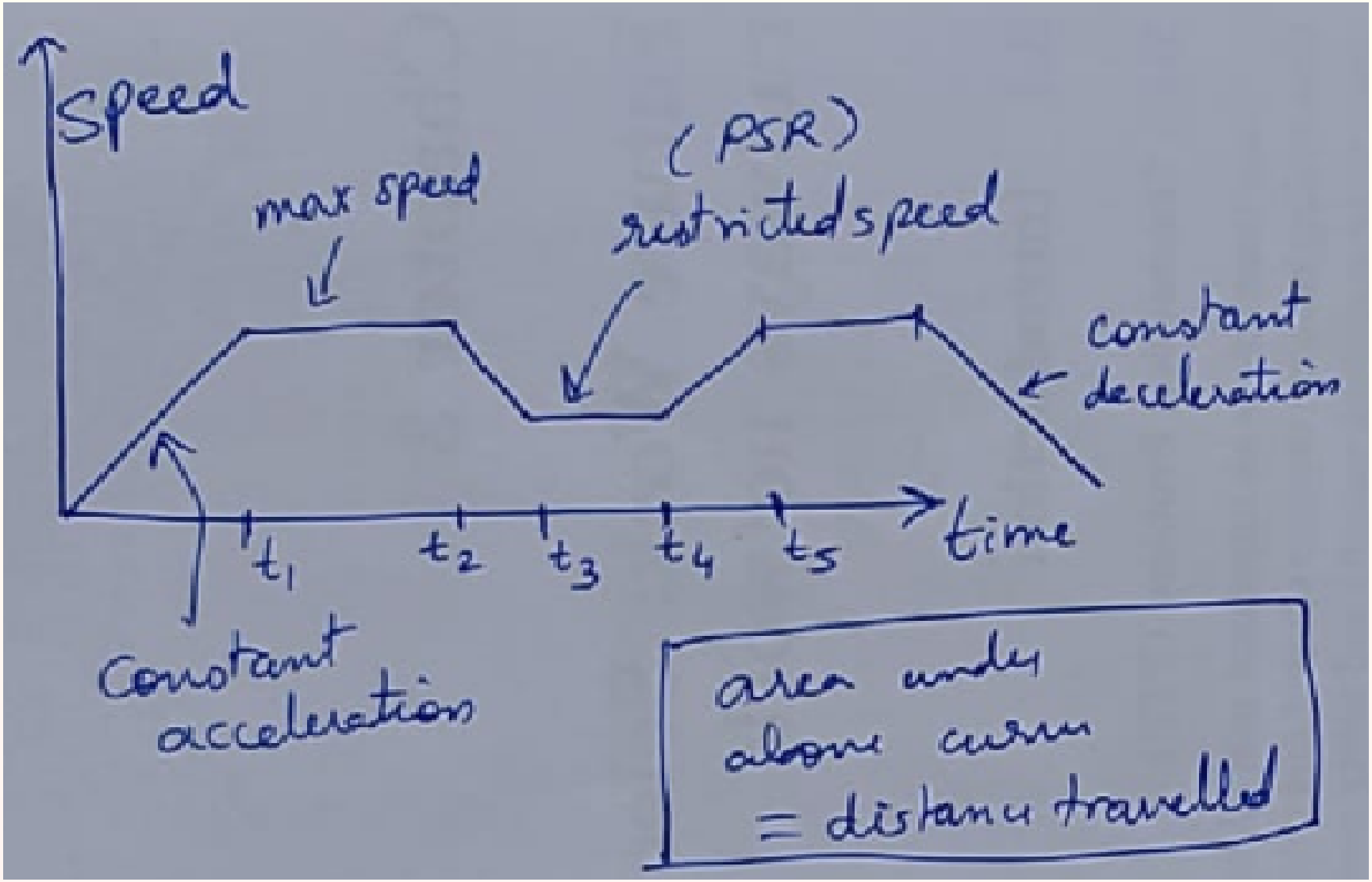


Figure: Speed vs time (more intuitive)

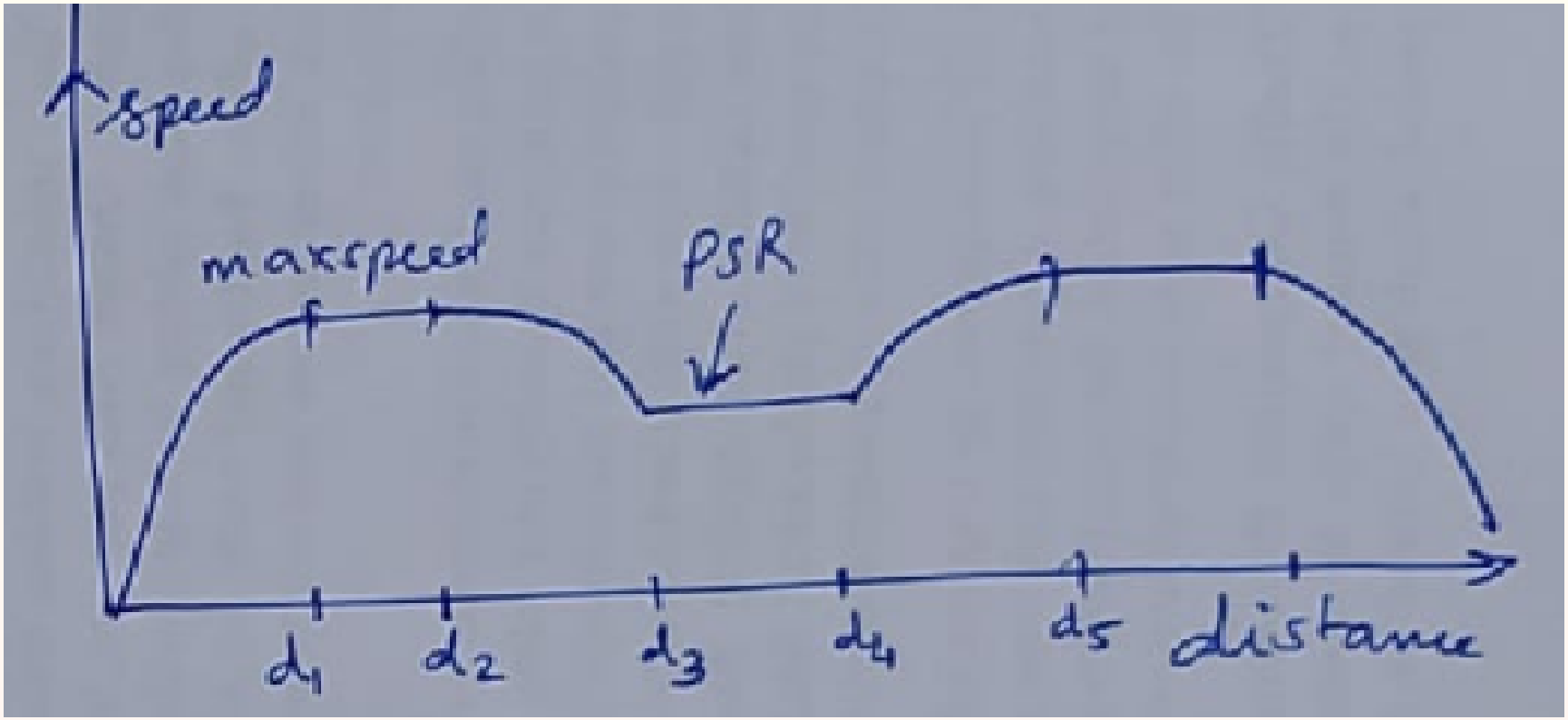
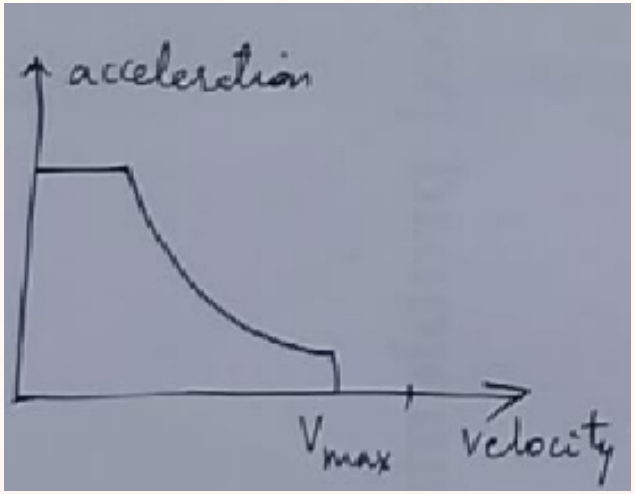


Figure: Speed vs distance (less intuitive, for timetabling)

In practice: acceleration is not constant (w.r.t. speed)

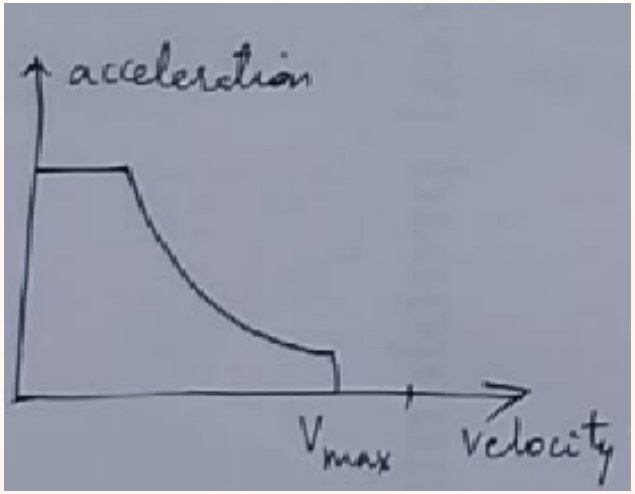
- Accln is lower at higher speeds (due to constant power reasons)
- Slope of speed vs time: decreases as speed increases



In summary: for more accuracy, need sophisticated tools

In practice: acceleration is not constant (w.r.t. speed)

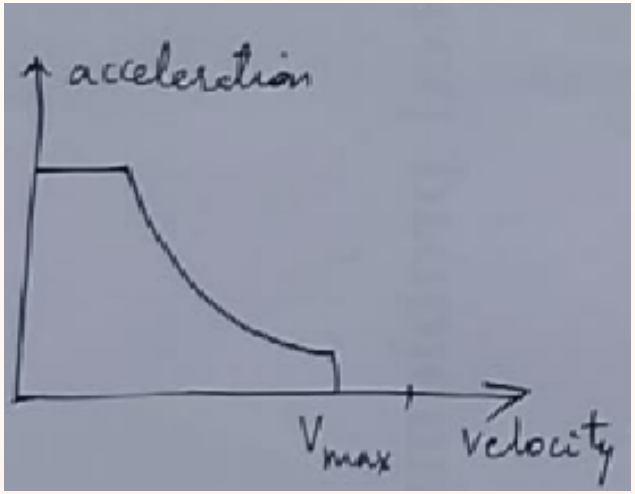
- Accln is lower at higher speeds (due to constant power reasons)
- Slope of speed vs time: decreases as speed increases



In summary: for more accuracy, need sophisticated tools
But need to start

In practice: acceleration is not constant (w.r.t. speed)

- Accln is lower at higher speeds (due to constant power reasons)
- Slope of speed vs time: decreases as speed increases

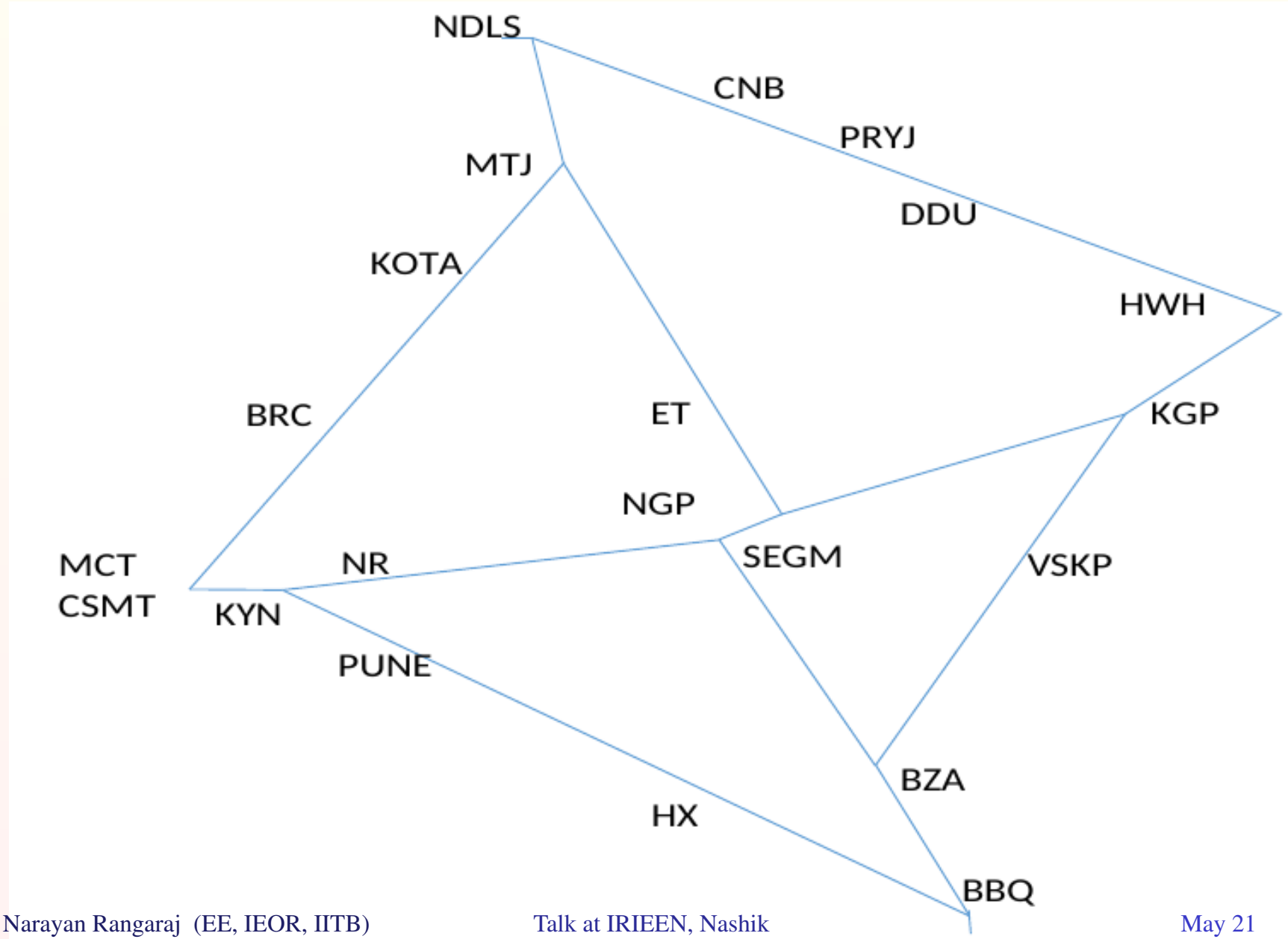


In summary: for more accuracy, need sophisticated tools
But need to start with simple tools/approximations, and (at least) for 'decision support' (semi-automation)

Network simulation tool

- Infrastructure up/down convention CANNOT be uniform in a network
- Multiple zero-mileposts since routes could overlap: much/little/none
- Possibly need to jump milepost distances (at junctions)
- Reversals are typical
- Break long train journey into parts, and simulate one by one (Navjeevan Exp)

GQD network: schematic (thanks to NR)



Integrated softwares are often clumsy

- Good design principle:

Integrated softwares are often clumsy

- Good design principle:
  "each tool does one small thing:

Integrated softwares are often clumsy

- Good design principle:
  “each tool does one small thing: and does it well"

Integrated softwares are often clumsy

- Good design principle:
  “each tool does one small thing: and does it well"
- Integrate these smaller tools to achieve a larger goal

Integrated softwares are often clumsy

- Good design principle:
  “each tool does one small thing: and does it well"
- Integrate these smaller tools to achieve a larger goal
- This is easier for debugging and for correction
- One improvement/change/correction won’t affect many other things.

Integrated softwares are often clumsy

- Good design principle:
  “each tool does one small thing: and does it well"
- Integrate these smaller tools to achieve a larger goal
- This is easier for debugging and for correction
- One improvement/change/correction won’t affect many other things.
- Additional features can be incorporated as extra options

Integrated softwares are often clumsy

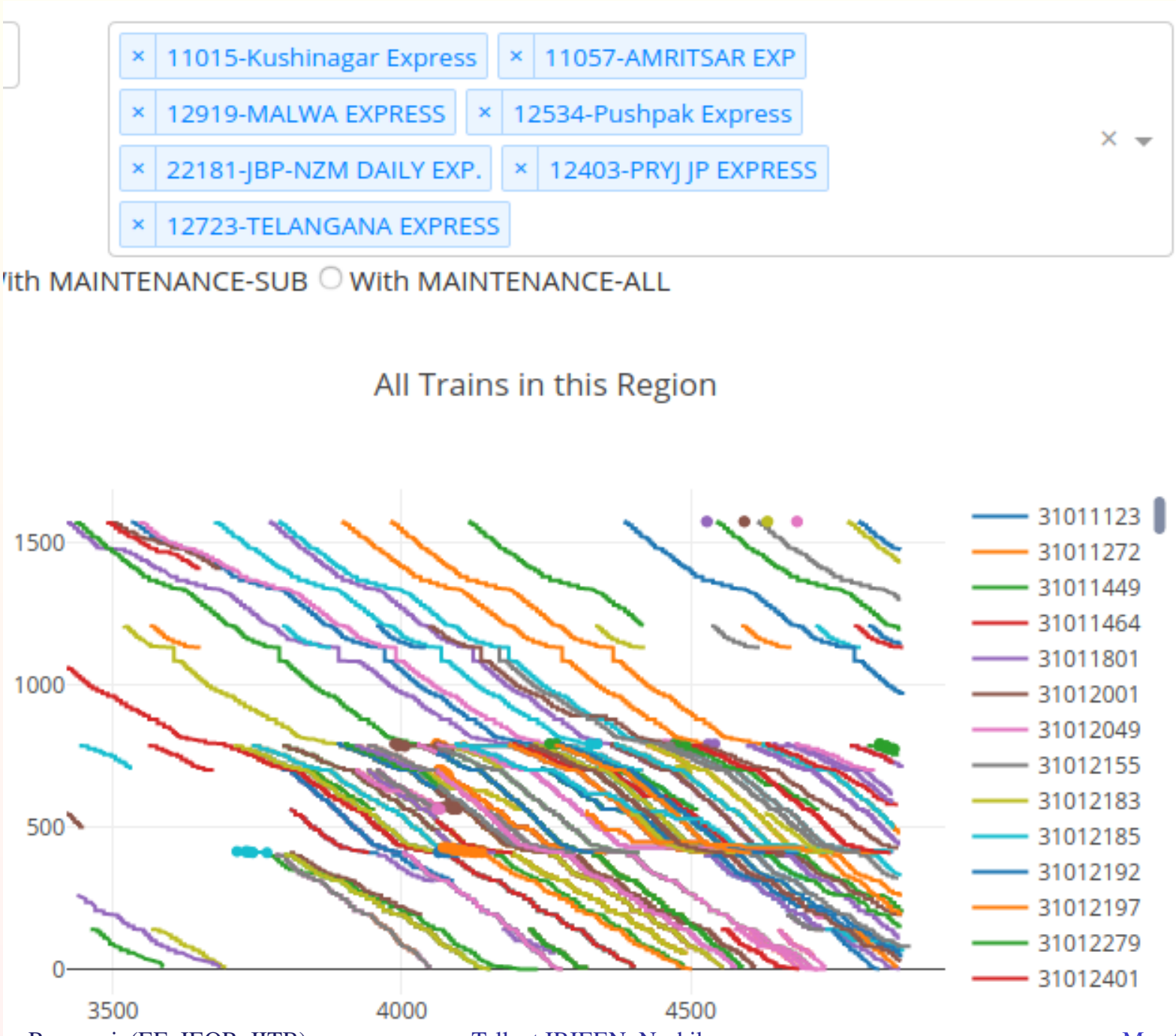
- Good design principle:
  “each tool does one small thing: and does it well"
- Integrate these smaller tools to achieve a larger goal
- This is easier for debugging and for correction
- One improvement/change/correction won’t affect many other things.
- Additional features can be incorporated as extra options

‘Software package’: more complete, integrated, stand-alone, with-GUI, etc.

Integrated softwares are often clumsy

- Good design principle:
  “each tool does one small thing: and does it well"
- Integrate these smaller tools to achieve a larger goal
- This is easier for debugging and for correction
- One improvement/change/correction won’t affect many other things.
- Additional features can be incorporated as extra options

‘Software package’: more complete, integrated, stand-alone, with-GUI, etc.
Usually, software company develops, owns and releases newer version


× 11015-Kushinagar Express
× 11057-AMRITSAR EXP
× 12919-MALWA EXPRESS
× 12534-Pushpak Express
× 22181-JBP-NZM DAILY EXP.
× 12403-PRYJ JP EXPRESS
× 12723-TELANGANA EXPRESS
ith MAINTENANCE-SUB
With MAINTENANCE-ALL
All Trains in this Region
1500
1000
500
0
3500
4000
4500
31011123
31011272
31011449
31011464
31011801
31012001
31012049
31012155
31012183
31012185
31012192
31012197
31012279
31012401

Trade-off between accuracy and elaborate details

Realistic-outcome, accuracy, elaborate-details vs quick-outcomes

Trade-off between accuracy and elaborate details

Realistic-outcome, accuracy, elaborate-details vs quick-outcomes

- To ensure simulation is not over-simplistic, need elaborate details of infrastructure, and train-running

Trade-off between accuracy and elaborate details

Realistic-outcome, accuracy, elaborate-details vs quick-outcomes

- To ensure simulation is not over-simplistic, need elaborate details of infrastructure, and train-running
- Often, indepth study difficult due to the need for data

Trade-off between accuracy and elaborate details

Realistic-outcome, accuracy, elaborate-details vs quick-outcomes

- To ensure simulation is not over-simplistic, need elaborate details of infrastructure, and train-running
- Often, indepth study difficult due to the need for data
- Realistic studies would inevitably require elaborate details

Trade-off between accuracy and elaborate details

Realistic-outcome, accuracy, elaborate-details vs quick-outcomes

- To ensure simulation is not over-simplistic, need elaborate details of infrastructure, and train-running
- Often, indepth study difficult due to the need for data
- Realistic studies would inevitably require elaborate details
- No shortcuts or 'low-hanging-fruits' remaining
- A continuous/long-term engagement needed for any fruitful outcome
- IR too complex for ready-made off-the-shelf solutions

Recent railway projects completed/ongoing

- ZBTT: only GQD routes: daily coaching-train paths and daily freight paths
- Ahmedabad junction: simulation based congestion study: 2019
- RDSO: section simulator: 2018
- Use of section simulator for Niti-Aayog: short study: 2017
- Use of section simulator for ALD study: for RITES Ltd: 2018
- CR/WR locals: security personnel assignment for late-night travel in ladies-compartments (9pm to 6am): 2017

Recent railway projects completed/ongoing

- ZBTT: only GQD routes: daily coaching-train paths and daily freight paths
- Ahmedabad junction: simulation based congestion study: 2019
- RDSO: section simulator: 2018
- Use of section simulator for Niti-Aayog: short study: 2017
- Use of section simulator for ALD study: for RITES Ltd: 2018
- CR/WR locals: security personnel assignment for late-night travel in ladies-compartments (9pm to 6am): 2017
- WR/CR: crew allotment and harbour line timetable: 2019
- Several other usages of this simulator: before 2016: kept in the link

Recent railway projects completed/ongoing

- ZBTT: only GQD routes: daily coaching-train paths and daily freight paths
- Ahmedabad junction: simulation based congestion study: 2019
- RDSO: section simulator: 2018
- Use of section simulator for Niti-Aayog: short study: 2017
- Use of section simulator for ALD study: for RITES Ltd: 2018
- CR/WR locals: security personnel assignment for late-night travel in ladies-compartments (9pm to 6am): 2017
- WR/CR: crew allotment and harbour line timetable: 2019
- Several other usages of this simulator: before 2016: kept in the link

Machinery/tools

- Java and Python: free and open source
- Free and Open-Source (FOSS): ‘independence’

Machinery/tools

- Java and Python: free and open source
- Free and Open-Source (FOSS): ‘independence’
- Helps to use open-source tools: else exorbitant (non-academic) prices

Machinery/tools

- Java and Python: free and open source
- Free and Open-Source (FOSS): ‘independence’
- Helps to use open-source tools: else exorbitant (non-academic) prices (and strings on usage)

Machinery/tools

- Java and Python: free and open source
- Free and Open-Source (FOSS): ‘independence’
- Helps to use open-source tools: else exorbitant (non-academic) prices (and strings on usage)
- Gurobi as the solver: Gurobi’s limited version is free
- IPOPT/COIN-OR (open-source and state of the art solvers): will shift soon to this

Machinery/tools

- Java and Python: free and open source
- Free and Open-Source (FOSS): ‘independence’
- Helps to use open-source tools: else exorbitant (non-academic) prices (and strings on usage)
- Gurobi as the solver: Gurobi’s limited version is free
- IPOPT/COIN-OR (open-source and state of the art solvers): will shift soon to this
- Commercial software: dependence: cannot install on many computers

Machinery/tools

- Java and Python: free and open source
- Free and Open-Source (FOSS): ‘independence’
- Helps to use open-source tools: else exorbitant (non-academic) prices (and strings on usage)
- Gurobi as the solver: Gurobi’s limited version is free
- IPOPT/COIN-OR (open-source and state of the art solvers): will shift soon to this
- Commercial software: dependence: cannot install on many computers
- Software made in FOSS can be modified to yield valuable statistics for analysis

Ready-made/proprietary vs 'home-grown' software

- IR: complex and unique challenges
- Other railways elsewhere: have their own challenges
- Each software needs significant local customization

Ready-made/proprietary vs 'home-grown' software

- IR: complex and unique challenges
- Other railways elsewhere: have their own challenges
- Each software needs significant local customization
- For complex requirements: prudent to have our own slowly-grown 'home-grown' software

Ready-made/proprietary vs 'home-grown' software

- IR: complex and unique challenges
- Other railways elsewhere: have their own challenges
- Each software needs significant local customization
- For complex requirements: prudent to have our own slowly-grown 'home-grown' software
- Problem specifications: from our own 'shop-floor': for example, operations personnel
- Railways/Academia/Software-agency: combination inevitable

Zero-base TT efforts elsewhere: example

The New Dutch Timetable: The OR Revolution, 2009 paper in *Interfaces*

Paper abstract

In December 2006, Netherlands Railways introduced a completely new timetable. Its objective was to facilitate the growth of passenger and freight transport on a highly utilized railway network, and improve the robustness of the timetable resulting in less train delays in the operation. Further adjusting the existing timetable constructed in 1970 was not option anymore, because further growth would then require significant investments in the rail infrastructure.

Constructing a railway timetable from scratch for about 5,500 daily trains was a complex problem. To support this process, we generated several timetables using sophisticated operations research techniques, and finally selected and implemented one of these timetables. Furthermore, because rolling-stock and crew costs are principal components of the cost of a passenger railway operator, we used innovative operations research tools to devise efficient schedules for these two resources.

The new resource schedules and the increased number of passengers resulted

Authors/affiliations (of the famous paper in Interfaces)

The New Dutch Timetable: The OR Revolution, 2009 paper in *Interfaces*
Authors: Kroon, Huisman, Abbink, Fioole, Ybema, Maroti, Schrijver, Steenbeek, Fischetti
Affiliations: railway personnel/software-firm/academia

- Department of Logistics, Netherlands Railways: Kroon, Huisman, Abbink, Fioole, Ybema
- Rotterdam School of Management, Erasmus University: Kroon, Maroti
- Econometric Institute, Erasmus University Rotterdam: Huisman
- CWI and University of Amsterdam: Schrijver
- Safiro Software Solutions: Steenbeek
- University of Padova Italy: Fischetti

Authors/affiliations (of the famous paper in Interfaces)

The New Dutch Timetable: The OR Revolution, 2009 paper in *Interfaces*
Authors: Kroon, Huisman, Abbink, Fioole, Ybema, Maroti, Schrijver, Steenbeek, Fischetti
Affiliations: railway personnel/software-firm/academia

- Department of Logistics, Netherlands Railways: Kroon, Huisman, Abbink, Fioole, Ybema
- Rotterdam School of Management, Erasmus University: Kroon, Maroti
- Econometric Institute, Erasmus University Rotterdam: Huisman
- CWI and University of Amsterdam: Schrijver
- Safiro Software Solutions: Steenbeek
- University of Padova Italy: Fischetti

Similar efforts in Germany: Narayan Rangaraj's collaborators

Other projects involving development of tools

Together with collaborators, supervised/developed tools for:

- Fault diagnosis in nuclear reactor sensors (140 sensors, complex dynamics)
  BRNS project: implementation in Matlab/Scilab,

Other projects involving development of tools

Together with collaborators, supervised/developed tools for:

- Fault diagnosis in nuclear reactor sensors (140 sensors, complex dynamics)
  BRNS project: implementation in Matlab/Scilab, in 2nd phase: Python

Other projects involving development of tools

Together with collaborators, supervised/developed tools for:

- Fault diagnosis in nuclear reactor sensors (140 sensors, complex dynamics)
  BRNS project: implementation in Matlab/Scilab, in 2nd phase: Python
- Joint Admission for M.Sc.: allocation program (ties/reservation/multiple-papers/multiple-rounds/transparency):
  3500 M.Sc. seats across IITs (amongst $\sim$7k JAM-qualified candidates)

Other projects involving development of tools

Together with collaborators, supervised/developed tools for:

- Fault diagnosis in nuclear reactor sensors (140 sensors, complex dynamics)
  BRNS project: implementation in Matlab/Scilab, in 2nd phase: Python
- Joint Admission for M.Sc.: allocation program (ties/reservation/multiple-papers/multiple-rounds/transparency):
  3500 M.Sc. seats across IITs (amongst $\sim$7k JAM-qualified candidates)
- Final project/position allotment for MTech/PhD candidates: EE, IITB

Other projects involving development of tools

Together with collaborators, supervised/developed tools for:

- Fault diagnosis in nuclear reactor sensors (140 sensors, complex dynamics)
  BRNS project: implementation in Matlab/Scilab, in 2nd phase: Python
- Joint Admission for M.Sc.: allocation program (ties/reservation/multiple-papers/multiple-rounds/transparency):
  3500 M.Sc. seats across IITs (amongst $\sim$7k JAM-qualified candidates)
- Final project/position allotment for MTech/PhD candidates: EE, IITB
- Admission shortlisting portal: semi-automation (in EE, IITB)

Other projects involving development of tools

Together with collaborators, supervised/developed tools for:

- Fault diagnosis in nuclear reactor sensors (140 sensors, complex dynamics)
  BRNS project: implementation in Matlab/Scilab, in 2nd phase: Python
- Joint Admission for M.Sc.: allocation program (ties/reservation/multiple-papers/multiple-rounds/transparency):
  3500 M.Sc. seats across IITs (amongst $\sim$7k JAM-qualified candidates)
- Final project/position allotment for MTech/PhD candidates: EE, IITB
- Admission shortlisting portal: semi-automation (in EE, IITB)
  About 400 candidates (and parents) can leave a day early (to home/other-IITs)

Other projects involving development of tools

Together with collaborators, supervised/developed tools for:

- Fault diagnosis in nuclear reactor sensors (140 sensors, complex dynamics)
  BRNS project: implementation in Matlab/Scilab, in 2nd phase: Python
- Joint Admission for M.Sc.: allocation program (ties/reservation/multiple-papers/multiple-rounds/transparency):
  3500 M.Sc. seats across IITs (amongst $\sim$7k JAM-qualified candidates)
- Final project/position allotment for MTech/PhD candidates: EE, IITB
- Admission shortlisting portal: semi-automation (in EE, IITB)
  About 400 candidates (and parents) can leave a day early (to home/other-IITs)
- Interview committee allotment (in EE, IITB): from 4 hrs $\searrow$ 1 min
  About 150 candidates (and parents) can leave a day early (to home/other-IITs)

Other projects involving development of tools

Together with collaborators, supervised/developed tools for:

- Fault diagnosis in nuclear reactor sensors (140 sensors, complex dynamics)
  BRNS project: implementation in Matlab/Scilab, in 2nd phase: Python
- Joint Admission for M.Sc.: allocation program (ties/reservation/multiple-papers/multiple-rounds/transparency):
  3500 M.Sc. seats across IITs (amongst $\sim$7k JAM-qualified candidates)
- Final project/position allotment for MTech/PhD candidates: EE, IITB
- Admission shortlisting portal: semi-automation (in EE, IITB)
  About 400 candidates (and parents) can leave a day early (to home/other-IITs)
- Interview committee allotment (in EE, IITB): from 4 hrs $\searrow$ 1 min
  About 150 candidates (and parents) can leave a day early (to home/other-IITs)

Summary: types of tools

- Constructive allocation tools: ensure constraints are satisfied
- Allocation based on 'solvers': satisfy all constraints (feasible solutions),

Summary: types of tools

- Constructive allocation tools: ensure constraints are satisfied
- Allocation based on 'solvers': satisfy all constraints (feasible solutions), search for 'best' feasible solution

Summary: types of tools

- Constructive allocation tools: ensure constraints are satisfied
- Allocation based on 'solvers': satisfy all constraints (feasible solutions), search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet

Summary: types of tools

- Constructive allocation tools: ensure constraints are satisfied
- Allocation based on 'solvers': satisfy all constraints (feasible solutions), search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet
- Simulation: for complex scenarios/rules/constraints

Summary: types of tools

- Constructive allocation tools: ensure constraints are satisfied
- Allocation based on 'solvers': satisfy all constraints (feasible solutions), search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet
- Simulation: for complex scenarios/rules/constraints
- Analysis: tool extracts statistics from simulation or other data
- Visualization tools: to ratify data
  and also view analysis-based statistics, aid further tool-usage

Summary: types of tools

- Constructive allocation tools: ensure constraints are satisfied
- Allocation based on 'solvers': satisfy all constraints (feasible solutions), search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet
- Simulation: for complex scenarios/rules/constraints
- Analysis: tool extracts statistics from simulation or other data
- Visualization tools: to ratify data
  and also view analysis-based statistics, aid further tool-usage
- Semi-automate: provide portal/interface to simplify complex problems
  portal takes inputs manually,
  after showing statistics, awaits further inputs:

Summary: types of tools

- Constructive allocation tools: ensure constraints are satisfied
- Allocation based on 'solvers': satisfy all constraints (feasible solutions), search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet
- Simulation: for complex scenarios/rules/constraints
- Analysis: tool extracts statistics from simulation or other data
- Visualization tools: to ratify data
  and also view analysis-based statistics, aid further tool-usage
- Semi-automate: provide portal/interface to simplify complex problems
  portal takes inputs manually,
  after showing statistics, awaits further inputs: 'decision support tools'

Summary: types of tools

- Constructive allocation tools: ensure constraints are satisfied
- Allocation based on 'solvers': satisfy all constraints (feasible solutions), search for 'best' feasible solution
- Validation/checker tools: easily checked in spreadsheet
- Simulation: for complex scenarios/rules/constraints
- Analysis: tool extracts statistics from simulation or other data
- Visualization tools: to ratify data
  and also view analysis-based statistics, aid further tool-usage
- Semi-automate: provide portal/interface to simplify complex problems
  portal takes inputs manually,
  after showing statistics, awaits further inputs: 'decision support tools'
- Operations persons are able to specify/describe the problem where automation is possible
- With a tool, one can then dwell on complex decision making
  (since routine-calculation is now automated/semi-automated)

Summary: need to shift to automated/semi-automated tools

- Need to shift to modern tools for
  - self-growth (for ourselves remaining relevant over next few decades)
  - system productivity/efficiency
- Tools that are 'home-grown' and in FOSS allow complete flexibility/independence and customization
- Describing/formulating the specs of the tool:
  - from on-field/operations personnel

Summary: need to shift to automated/semi-automated tools

- Need to shift to modern tools for
  - self-growth (for ourselves remaining relevant over next few decades)
  - system productivity/efficiency
- Tools that are 'home-grown' and in FOSS allow complete flexibility/independence and customization
- Describing/formulating the specs of the tool:
  - from on-field/operations personnel (primarily)

Summary: need to shift to automated/semi-automated tools

- Need to shift to modern tools for
  - self-growth (for ourselves remaining relevant over next few decades)
  - system productivity/efficiency
- Tools that are 'home-grown' and in FOSS allow complete flexibility/independence and customization
- Describing/formulating the specs of the tool:
  - from on-field/operations personnel (primarily)
  - from anybody interested in the area

Questions, contact details

Questions ?

Thank you

Contact details (Madhu Belur):

99874 66 279, belur@iitb.ac.in

This talk currently at
www.ee.iitb.ac.in/%7Ebelur/talks

Suburban (WR) crew-allotment conference paper
http://www.ee.iitb.ac.in/%7Ebelur/pdfs/c19or-dresden.pdf
See also slides by many other speakers also:
Workshop was about suburban-railways/metro operations planning (June 2019 workshop)
www.ee.iitb.ac.in/%7Ebelur/railways/workshop