# Simulation and Resource-Based Scheduling

Nathaniel Osgood

3-30-2004

# Topics

- Simulation Methods
  - Static vs dynamic scheduling
  - Scheduling granularity
  - GERT
  - Demos
  - Process interaction languages
  - Activity scanning languages
- Resource algorithms
  - Optimal
  - Heuristic
- Line of balance method

# Static vs. Dynamic Scheduling

- Static Scheduling
  - Dates known ahead of time
    - Coordination with stakeholders
    - Coordination with subcontractors
  - Coordination pre-planned (e.g. reservation system)
  - Can't take advantage of early finishes, arrivals, etc.
- Dynamic Scheduling ("Wait and See")
  - Dynamic coordination takes time (queuing)
  - Prevents needless waiting if finish early
- Both get done!

# Topics

- Simulation Methods
  - ✓ Static vs dynamic scheduling
  - Scheduling granularity
  - GERT
  - Demos
  - Process interaction languages
  - Activity scanning languages
- Resource algorithms
  - Optimal
  - Heuristic
- Line of balance method

# Granularity of Scheduling

- Fine-grained (particular crews, indiv.) advantages
  - Can reason about
    - Crew productivity
    - Balancing workload among crews
    - Distance traveled for particular crews
- Coarser-grained advantages
  - Combinatorially more challenging
  - May inhibit creativity in dynamic scheduling
- Generally, leave *dynamic* (or "quasi-dynamic") factors that don't consider statically

# Topics

- Simulation Methods
  - ✓ Static vs dynamic scheduling
  - ✓ Scheduling granularity
  - GERT
  - Demos
  - Process interaction languages
  - Activity scanning languages
- Resource algorithms
  - Optimal
  - Heuristic
- Line of balance method

# Major Types of Simulation

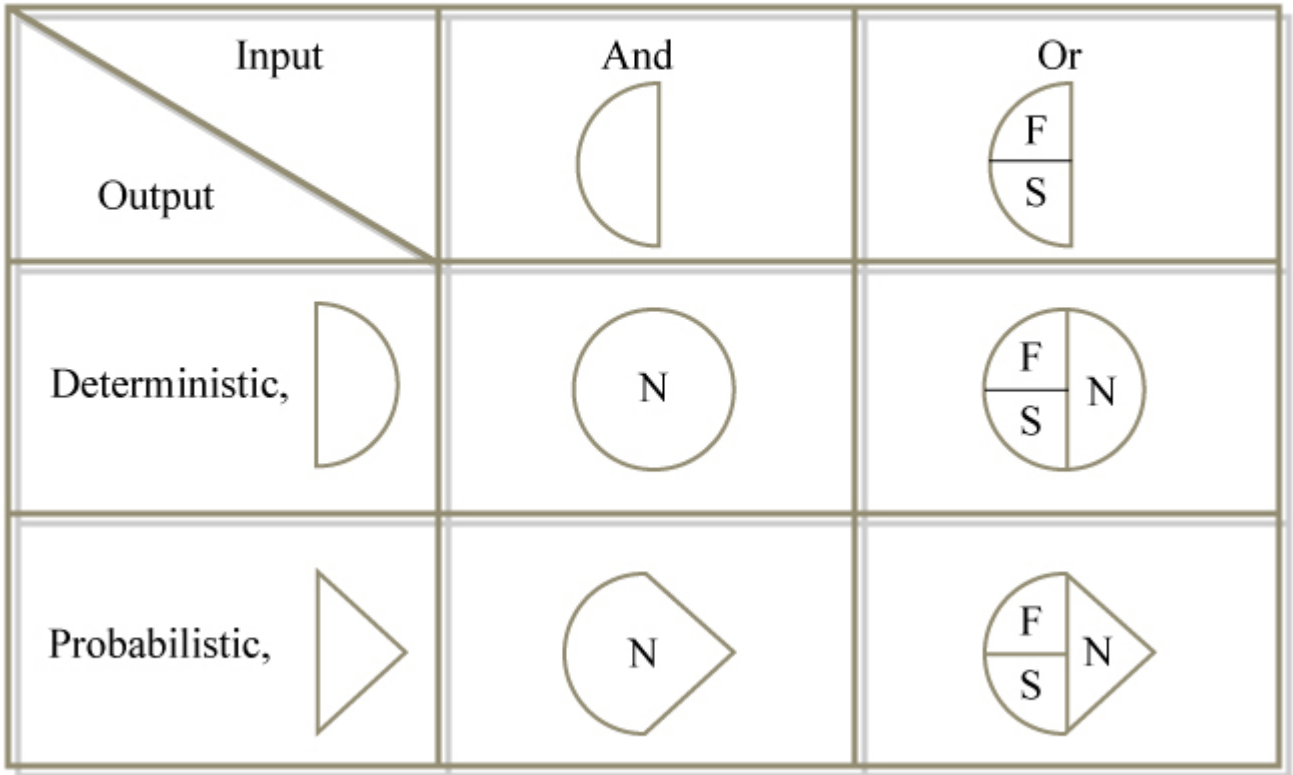
- Discussed here: Discrete Event Simulation
  - Oldest:  GERT/Q-GERT (Pritsker)
  - Common representation: Activity Cycle Diagrams
    - Transaction-based ("Process interaction") approaches
      - SLAM
      - GPSS
    - Activity-scanning approaches
      - CYCLONE
      - STROBOSCOPE
- Not discussed today: System Dynamics
- NB: graphical rep. & implementation may differ

# GERT Basics

- Builds on Activity-on-Arrow diagrams
- Discrete semantics
  - "Transactions" ("entities") flow through system
  - Fire off additional transactions
- Transactions secure resources for processing
  - Must wait until needed resources are available
- Probabilistic
  - Durations
  - Branching & Looping
    - Time no longer linearly increasing to right!

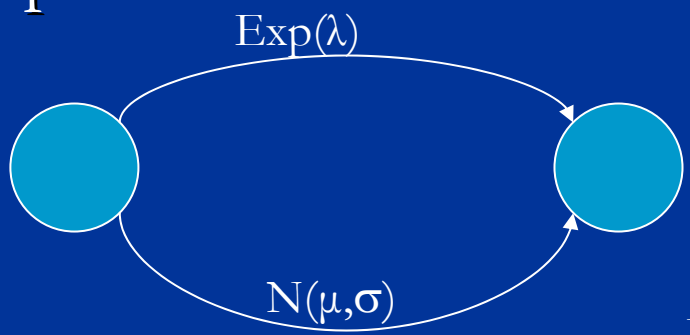
# Most Common Q-GERT Node

A Form of Basic GERT Notation

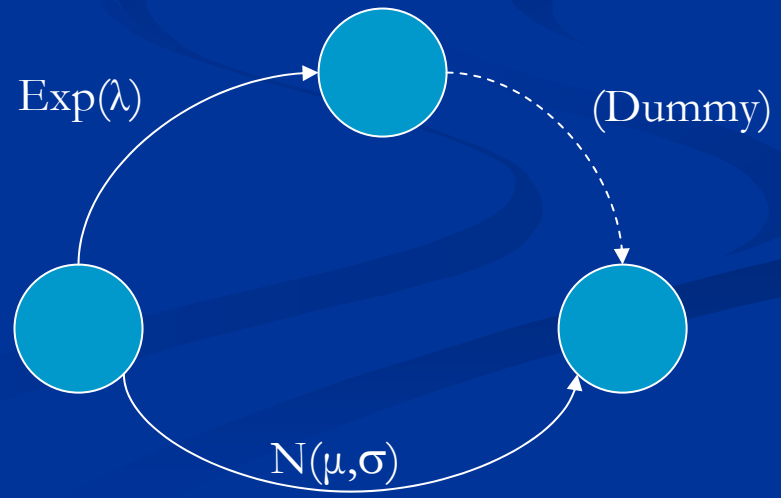


# Arrows

- Represent *activities*
- Sometimes use dummies, sometimese not
  - Representation 1:



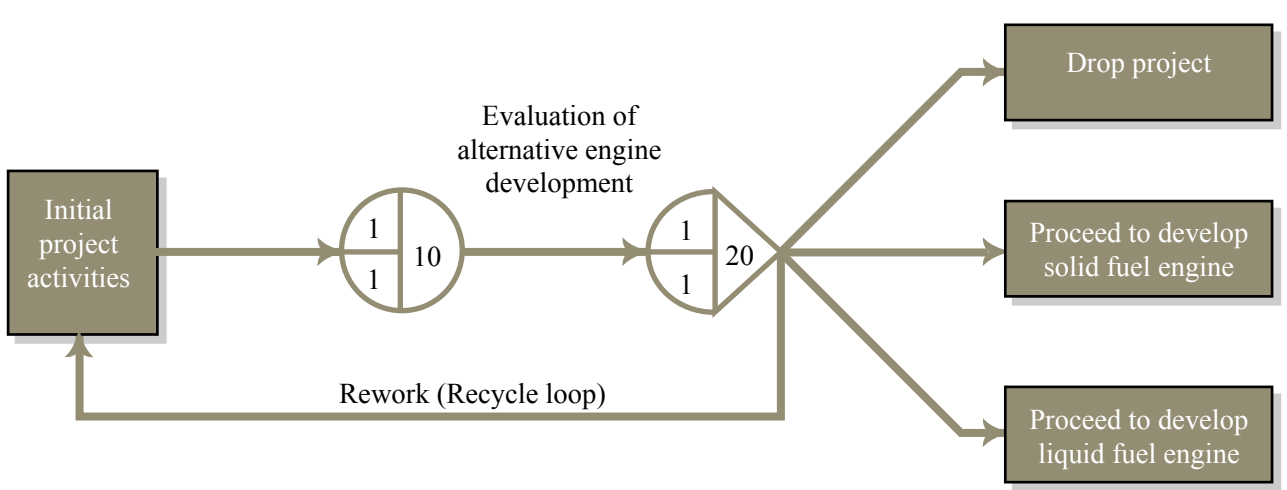

  - Representation 2:



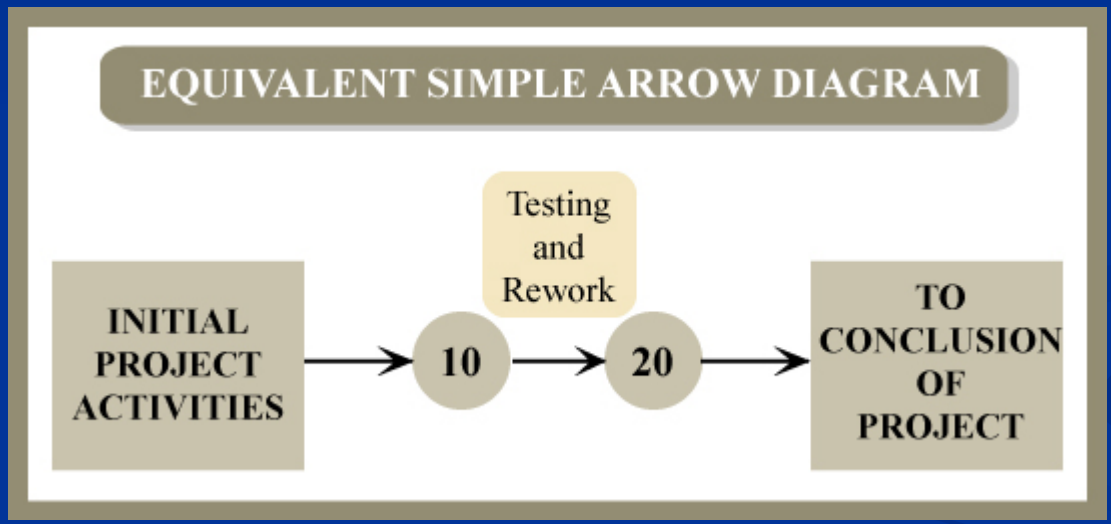
# GERT
# Looping/Branching Constructs

- Branching
  - Deterministic (like CPM/PERT)
  - Probabilistic
- Rework Cycle

Example of Probabilistic Branching and Looping-no Equivalent Arrow Diagram Possible.



# Corresponding Aggregate CPM Representation



# Additional GERT Node Types

- Source/Sink nodes
- Statistics nodes
  - First/all realizations
  - Interarrival time
  - Time interval from mark node
  - Etc.
- Mark nodes
- Diversity in realization types, etc.

# Q-GERT: Introduces Queues

- Delivers entities to outgoing activities
  - Outgoing activities are associated with # of servers
    - # clients that can process in parallel
- Accumulates objects up to some capacity
  - Backs up if cannot process immediately
  - "Balks" if reaches capacity
- Combination elements to combine transactions
- Led to development of GASP, SLAM

# GERT Applications

- Offshore Oil Rigs (Granli)
- Alaska Pipeline
- Reservoir construction (Pena Mora)

# Construction Simulation Languages

- Equally powerful:  All "Turing Equivalent"
  - i.e. Any language can simulate any other
- Differ in terms of
  - Ease of use
  - Extensibility
  - Expressiveness for construction domain
    - How natural is it to describe construction scenarios?
- Two main framework:  process interaction and activity scanning

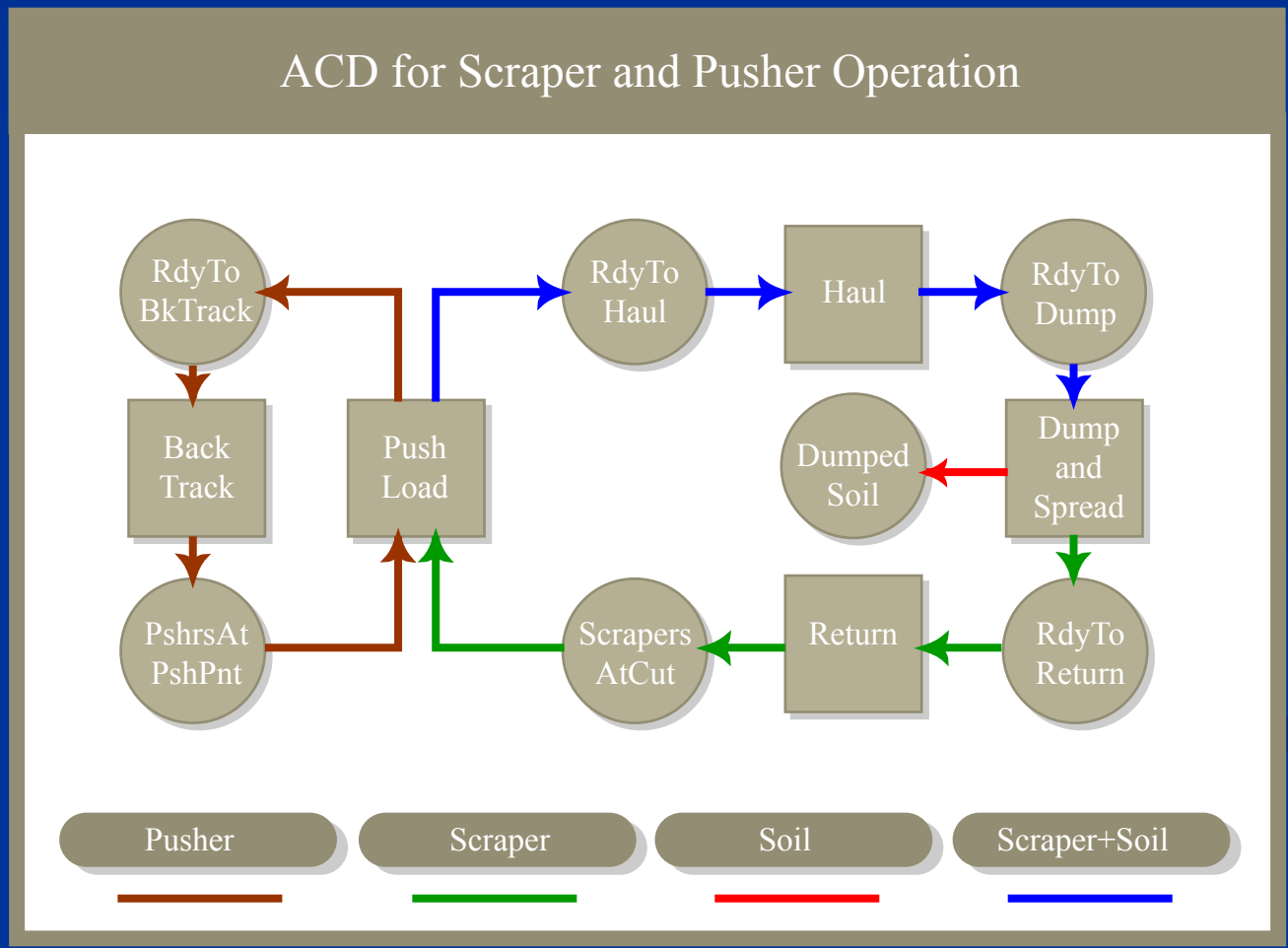
# Topics

- Simulation Methods
  - ✓ Static vs dynamic scheduling
  - ✓ Scheduling granularity
  - ✓ GERT
  - Demos
  - Process interaction languages
  - Activity scanning languages
- Resource algorithms
  - Optimal
  - Heuristic
- Line of balance method

# Activity Cycle Diagrams: Common Visual Representation

Rectangles: Activities

Circles: Queues (resources in particular state)



# Simulation Example: Excavation and Transporting

- Given
  - Front-end loader
    - Output: $o_{\text{front-end loader}}$
    - Instantaneous time between loads
  - Trucks
    - n vehicles
    - Capacity c
    - Load time $t_l$
    - Instantaneous dump time
    - Fully loaded speed $s_l$ , empty speed $s_e$
    - Distance to dumpsite d

$$o_{\text{trucks}} = \frac{nc}{\frac{d}{s_l} + \frac{d}{s_e}} = \frac{ncs_l s_e}{d(s_e + s_l)}$$

- Naïve productivity: $\min(o_{\text{front-end loader}}, o_{\text{trucks}})$

# Topics

- Simulation Methods
  - ✓ Static vs dynamic scheduling
  - ✓ Scheduling granularity
  - ✓ GERT
  - ✓ Demos
  - Process interaction languages
  - Activity scanning languages
- Resource algorithms
  - Optimal
  - Heuristic
- Line of balance method

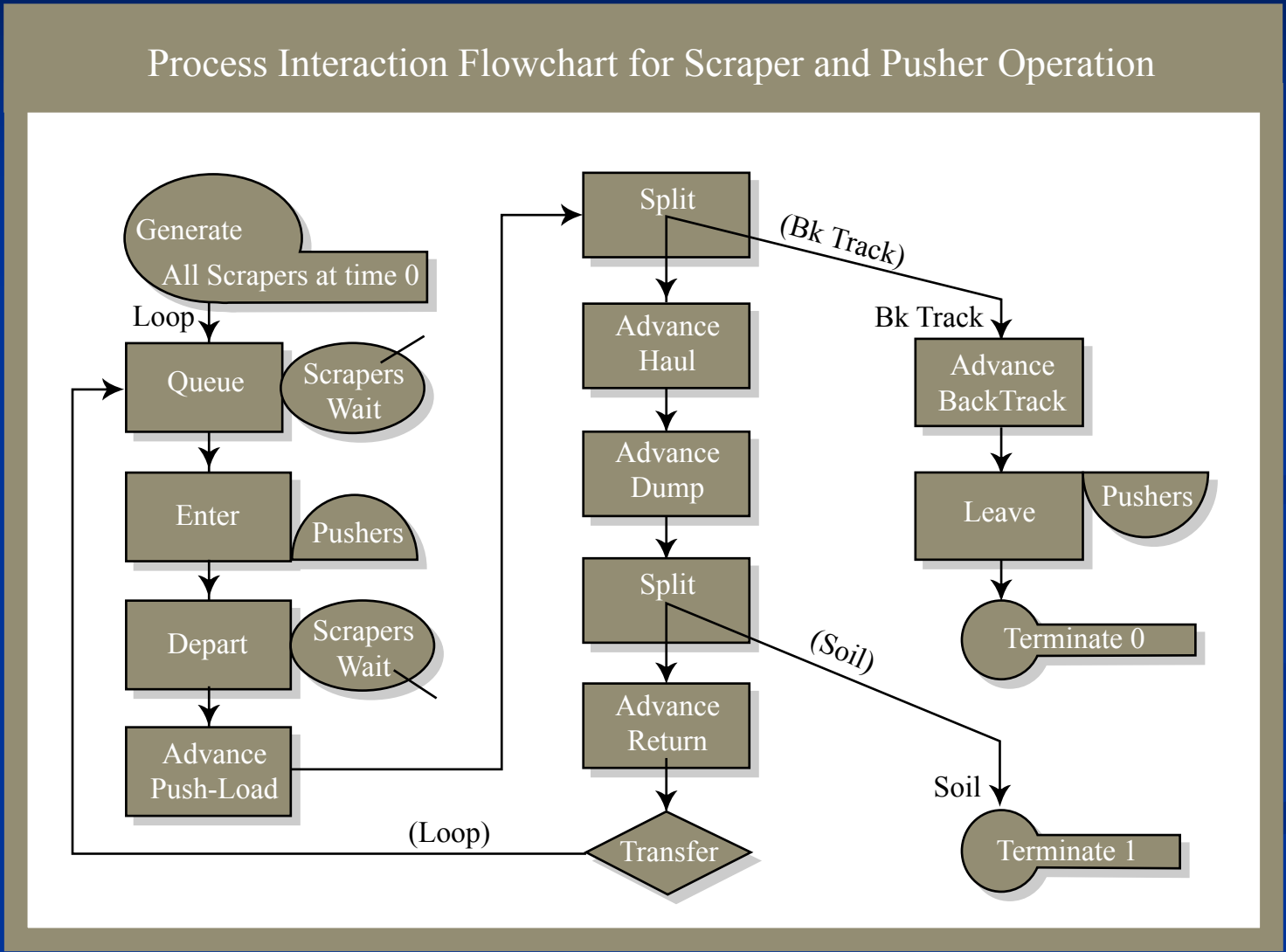
# Process Interaction Frameworks

- Examples: SLAM, GPSS, ProModel, SimScript, ModSim, etc.
- Based on flow of transactions through system
  - Capture and release resources ("machines")
- Good in "job shops" where have
  - Clear items moving through system
  - Resources temporarily claimed by items

# Process Interaction Shortcomings

- In construction, identifying what constitutes a "transaction" can be tricky
  - Awkward to represent other entities
- Changing perspective to other entities difficult
- Risk of deadlock over resources during simulation

# Process Interaction Representation



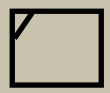
# Topics

- Simulation Methods
  - ✓ Static vs dynamic scheduling
  - ✓ Scheduling granularity
  - ✓ GERT
  - ✓ Demos
  - ✓ Process interaction languages
  - Activity scanning languages
- Resource algorithms
  - Optimal
  - Heuristic
- Line of balance method

# Activity Scanning Simulation Packages

- General purpose
- Representation similar to activity cycle diagrams
- Identify conditions to allow activities to proceed
- Construction-specific
  - CYCLONE
  - STROBOSCOPE

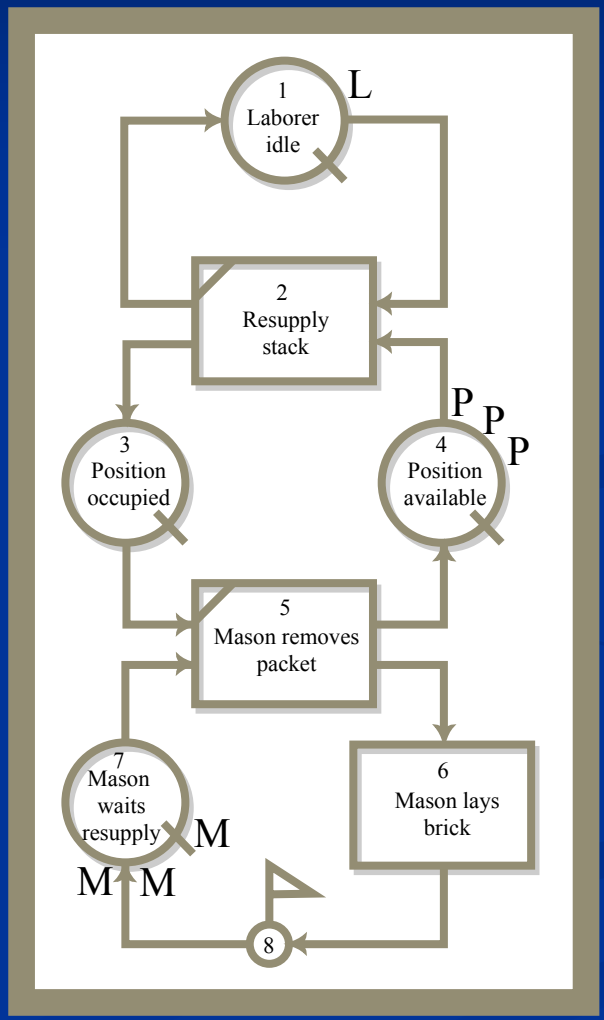
# CYCLONE Elements

| NAME | SYMBOL | FUNCTION |
|---|---|---|
| Combination (COMBI) Activity |  | This element is always preceded by Queue Nodes. Before it can commence, units must be available at each of the preceding Queue Nodes. If units are available, they are combined and processed through the activity. If units are available at some but not all of the preceding Queue Nodes, these units are delayed until the condition for the combination is met |
| Normal Activity |  | This is an activity similar to the COMBI. However, units arriving at this element begin processing immediately and are not delayed |
| Queue Node |  | This element precedes all COMBI activities and provides a location at which units are delayed pending combination. Delay statistics are measured at this element |
| Function Node |  | It is inserted into the model to perform special fucntion such as counting, consolidation, marking, and statistic collection |
| Accumulator |  | It is used to define the number of times the system cycles |
| Arc |  | Indicates the logical structure of the model and direction of entity flow |

# CYCLONE Brick Example Qualitative Description

- Labor:  3 masons, 1 laborer
- Scaffold can hold 3 pallets of 10 bricks each

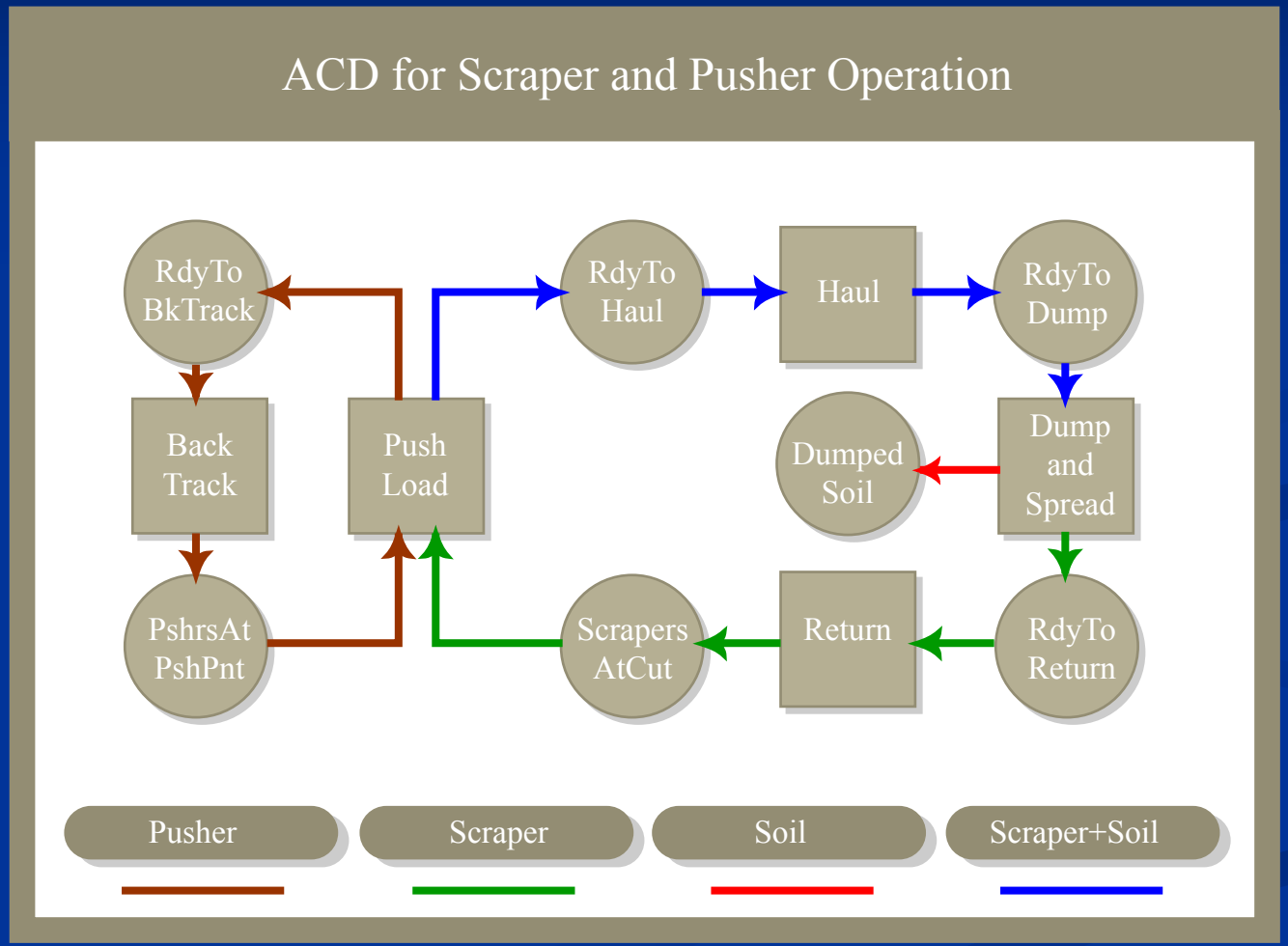
# CYCLONE Brick Example Model



# CYCLONE Brick Example Underlying Code

- NAME 'MASONRY' LEN 500 CYC 9
- NETWORK INPUT
- 1 QUE 'LAB IDLE'
- 2 COM SET 1 'RESUPPLY' FOL 1 3 PRE 1 4
- 3 QUE 'STACK OCCUP'
- 4 QUE 'STACK EMPTY'
- 5 COM SET 2 'PICKUP' FOL 4 6 PRE 3 7
- 6 NOR SET 3 'PLACE 10 BRK' FOL 8
- 7 QUE 'MASONS IDLE'
- 8 FUN COU QUA 10 FOL 7
- RESOURCE INPUT
- 1 'LABORER' AT 1
- 3 'POSITIONS' AT 4
- 3 'MASONS' AT 7
- DURATION INPUT
- SET 1 BETA 0.001 5.0 2.6 0.5
- SET 2 1
- SET 3 BETA 3 10 7 2.2
- ENDDATA

# ACD for Scrapers and Pushers



# Topics

- Simulation Methods
  - ✓ Static vs dynamic scheduling
  - ✓ Scheduling granularity
  - ✓ GERT
  - ✓ Demos
  - ✓ Process interaction languages
  - ✓ Activity scanning languages
- Resource algorithms
  - Optimal
  - Heuristic
- Line of balance method

# Recall Basic Steps of Network Methods

- Define activities from WBS work packages
- Estimate $, time, resources for each activity
- Define precedence relationships between activities
- Iterate
  - Perform CPM scheduling
    - In a sense, we are assuming "infinite resources" here!
  - Estimate time, cost, resource usage over project
  - If acceptable, terminate
  - If not acceptable, impose dependencies or added/reduced resources

# Resource Considerations

- Human resources most important
  - Time to procure
  - Difficult to release
  - Difficult to reuse on demand
  - Must consider different trades as different types of resources
- Equipment
  - Highly specialized equipment
  - Normal equipment relatively easy to procure on demand
  - Note constraints -- Permitting issues, etc!

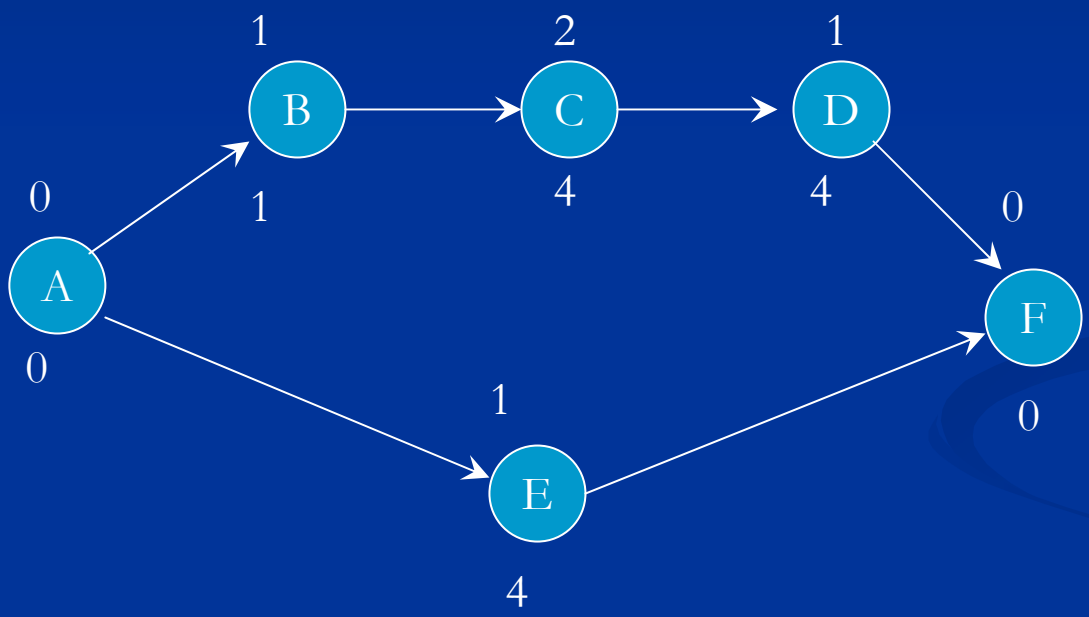
# Resource Algorithms

- Resource Leveling
  - Moving activities within float to minimize fluctuations
    - No change to schedule duration!
  - Especially important for human resources
  - No guarantee that falls within limits of available resources!
- Resource Scheduling ("Constrained-resource scheduling", "Limited Resource Allocation")
  - Scheduling resources within *constraints* with minimal extension of schedule time
- Key assumption: Holding individual *activity* resource assignments, durations *constant*

# Optimal Resource Scheduling/Leveling Methods

- Combinatorial problem
  - Computationally very expensive (NP-complete)
  - In principle, would need to compare all possible orderings of conflicting activities
  - Intuition: Lots of possible start/stop times within constraints
    - Typically too large to realistically enumerate
    - *Locally* best choice not necessarily *globally* best
- Approaches
  - Linear Programminig
  - Explicit Enumeration
  - "Branch and Bound" (constrained search)

# Example of Combinatorial Factors Resource Scheduling




LEGEND:

Required resources

E

Duration

Best local choice!

# Possible Schedules

```
B    CCCC
EEEECCCCDDDD
```

Duration: 12

```
     CCCC
BEEEECCCCDDDD
```

Duration: 13

```
 CCCC
BCCCCEEEEDDDD
```

Duration: 13

```
 CCCC
BCCCCDDDDEEEE
```

Duration: 13

```
     CCCC
EEEEBCCCCDDDD
```

Duration: 13

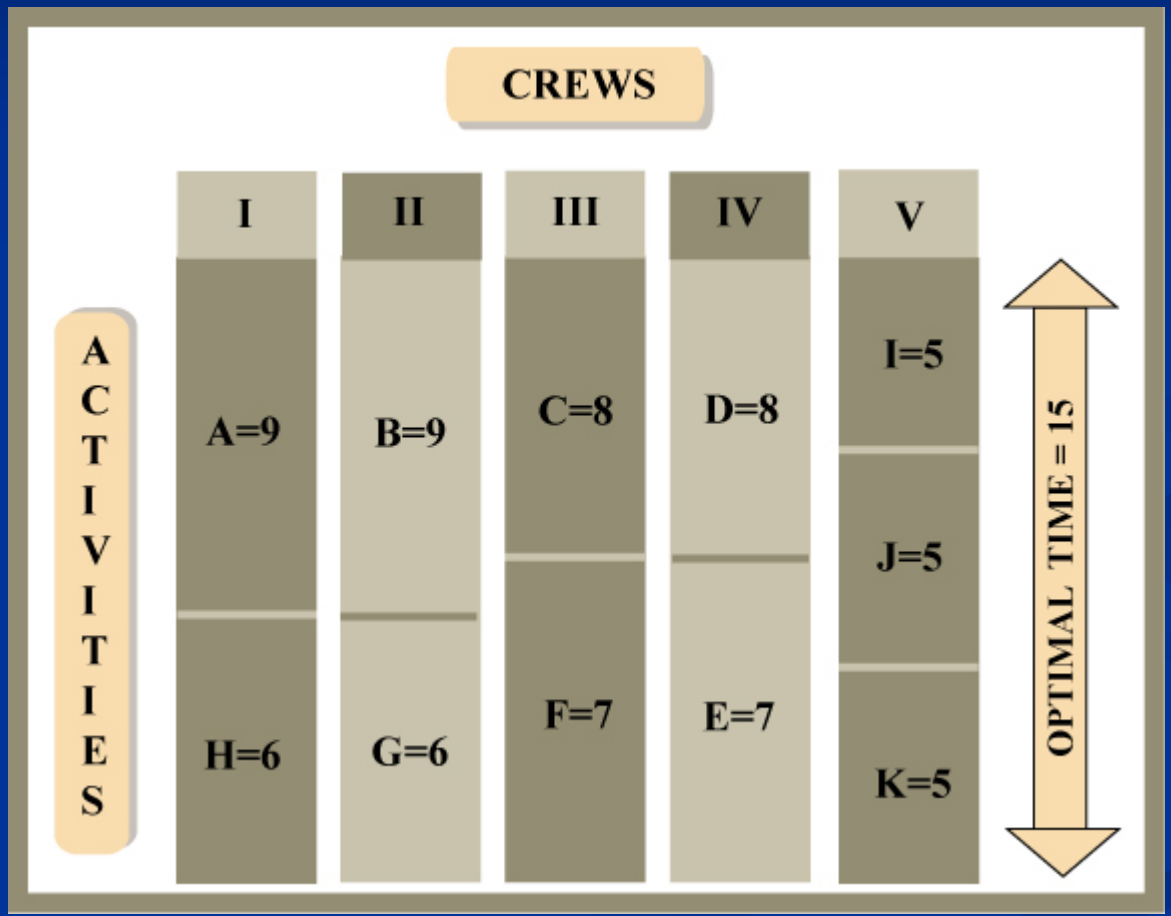
```
 CCCCEEEE
BCCCCDDDD
```

Duration: 9

# Detailed (Crew-Level Scheduling)

| Section | Work Duration |
|---|---|
| A | 9 |
| B | 9 |
| C | 8 |
| D | 8 |
| E | 7 |
| F | 7 |
| G | 6 |
| H | 6 |
| I | 6 |
| J | 5 |
| K | 5 |

# Assignment of Crews to Activities



# Integer Programming Formulation

- Section i, crew j
  - $x_{ij}$ = 1 if section i assigned to crew j, 0 otherwise
  - $t_i$ = time for section j
- Problem:
  - Minimize z subject to constraints

$$z \geq \sum_{i=1}^{n_{activities}} t_i x_{ij}$$

Need to account for assignment of time to resources

$$\sum_{i=j}^{n_{crews}} x_{ij} = 1$$

Only one crew assigned per resource

# Topics

- Simulation Methods
  - Static vs dynamic scheduling
  - Scheduling granularity
  - GERT
  - Demos
  - Process interaction languages
  - Activity scanning languages
- Resource algorithms
  - Optimal
  - Heuristic
- Line of balance method

# Heuristic Methods

- Use "rules of thumb" to get answer in acceptable time
- Typically reach local minimum
  - "Greedy" algorithms do what is locally – but not necessarily globally – best
- Typically use empirical evaluation to judge
- Sometimes highly problem-specific (e.g. SPAR) sometimes very general (branch & bound, simulated annealing, genetic algorithms, etc.)

# Heuristic Scheduling Approaches

- "Serial" methods: Schedule activity-by-activity
  - Consider prioritized activities in order
  - Schedule as early as possible
  - Less common
- "Parallel" methods: Schedule by timestep
  - Start with some initial schedule
  - For each timestep, decide which activities to delay
    - *Local* optimization: *Never reconsider activities in progress*
    - Based on characteristics of activities that would be active
  - Most commonly used

# Heuristic Resource Scheduling Methods: Typical Approach

- Assign priorities. Example metrics:
  - Shortest Task First
  - Most Resources First
  - Minimum Slack First
  - Most Critical Followers
  - Most Successors
- Wait until
  - Predecessors complete
  - Adequate resources available

# Empirical Studies

- Patterson and Davis (1975) compared multiple heuristics, in "serial" and "parallel" modes
  - Examples: Minimum late finish time, function of EF&LS, greatest resource demand, minimize resource "idling", shortest durations first, greatest # of activities, random activity selection, min. slack
  - Most effective:
    - Min slack (i.e. min late start)
    - Minimum late finish
  - Non-optimal 60% of time
  - Serial vs. parallel may be more important than rule

# Example *Leveling* Heuristic (Burgess)

- Sort activities in reverse order of precedence
- Assign activities to early start time
- Repeat until sum of squares of resource use =
  - For each activity in sorted order
    - Schedule at time that minimizes sum of squares of resource use (subject to precedence constraints)
- Repeat with different initial ordering if resource highly critical
- Adjust for any other factors omitted

# Example Heuristic Resource *Scheduling* Algorithm

1. Rank all resources from the most important to the least important
2. Set the scheduled start day for each activity to Early Start
3. Starting at $t = 0$, compute demand for resource $i$
4. If resource demand for $i$ is greater than availability, select the activity with the greatest Late Start and shift its start to $t + 1$
5. Repeat [4] until resource constraint at $t$ for resource $i$ is satisfied.
6. Repeat [4] for $t = t + 1$
7. Repeat [3] for resource $i + 1$

# Note on Prioritization of Algorithm

- Assuming no changing of already scheduled activities, delaying activities in order of late start is same as prioritizing in terms of slack
  - Same as minimizing extension to overall project duration
  - Selecting minimum slack shown empirically to outperform others
- Late finish-based selection also effective

# SPAR Algorithm

- Probabilistic assignment of jobs
  - Allows for finding multiple local minima
- Not simply "greedy": Reconsiders earlier decisions
- May change critical path (delay job or resources for crit)
- Linear time/resource tradeoff assumed
  - i.e. Constant person-days required for activity
- Critical resources considered for higher resource allocation
  - If necessary, borrow resources from previously scheduled jobs – but only if don't extend entire project
  - If absolutely necessary, delay other active jobs

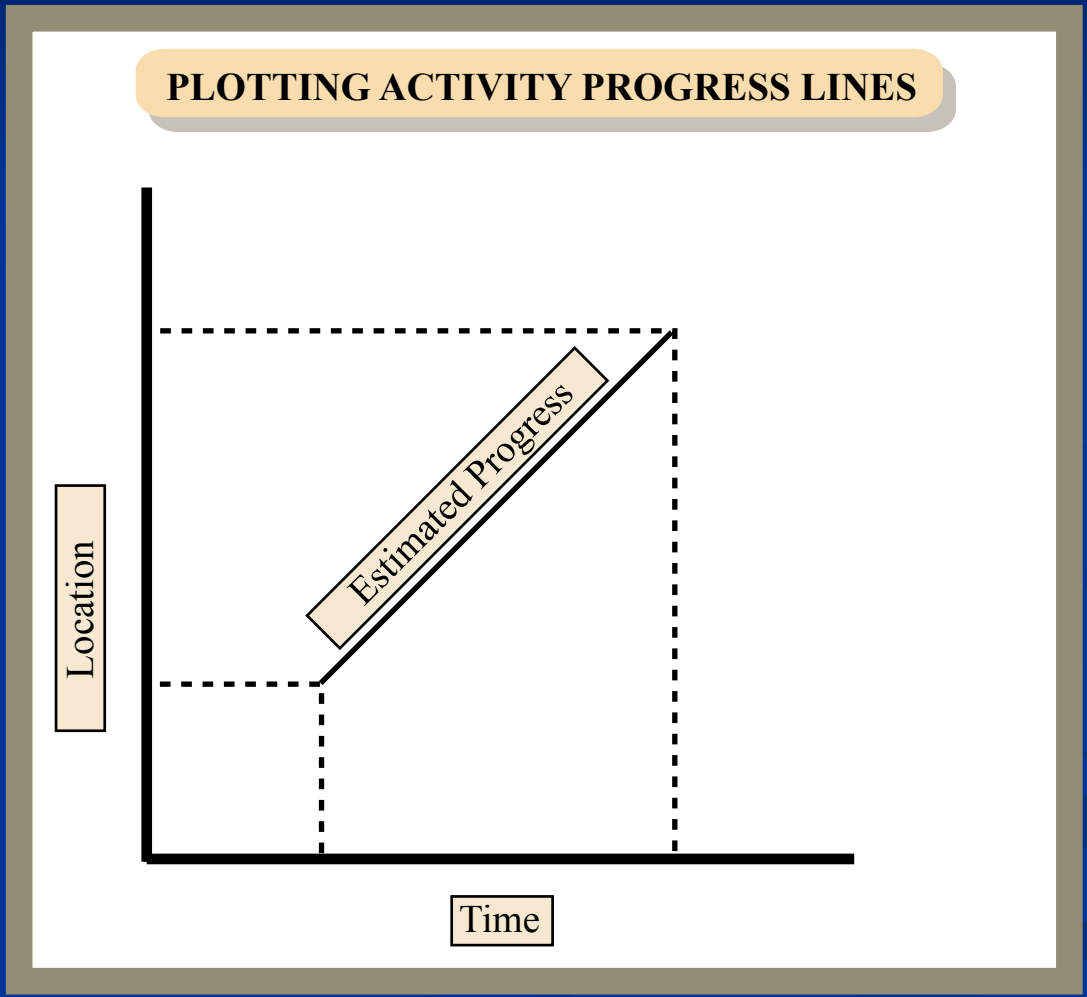
# Topics

- Simulation Methods
  - Static vs dynamic scheduling
  - Scheduling granularity
  - GERT
  - Demos
  - Process interaction languages
  - Activity scanning languages
- Resource algorithms
  - Optimal
  - Heuristic
- Line of balance method

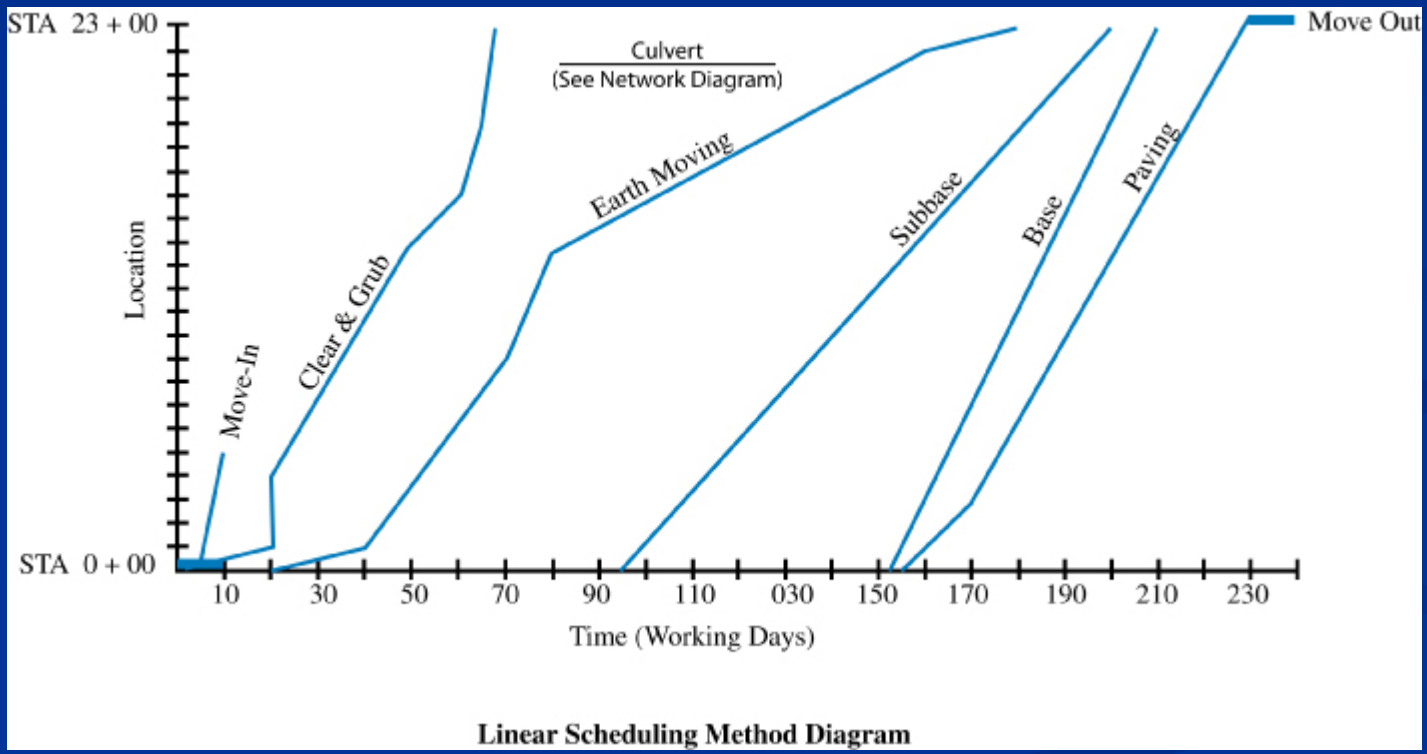
# Line-of-Balance Applicability

- Repetitive Activities
- Same Crew
- Rate of Progress
- Minimize Discontinuity by Crews
- Represented in the Network by
  - Continuous Activities
  - Repetitive Activities
- Use for planning
  - Crew reuse
  - Planning timing of work
  - Planning speed of work

# Plotting Activity Progress Lines



# Linear Scheduling Method Diagram

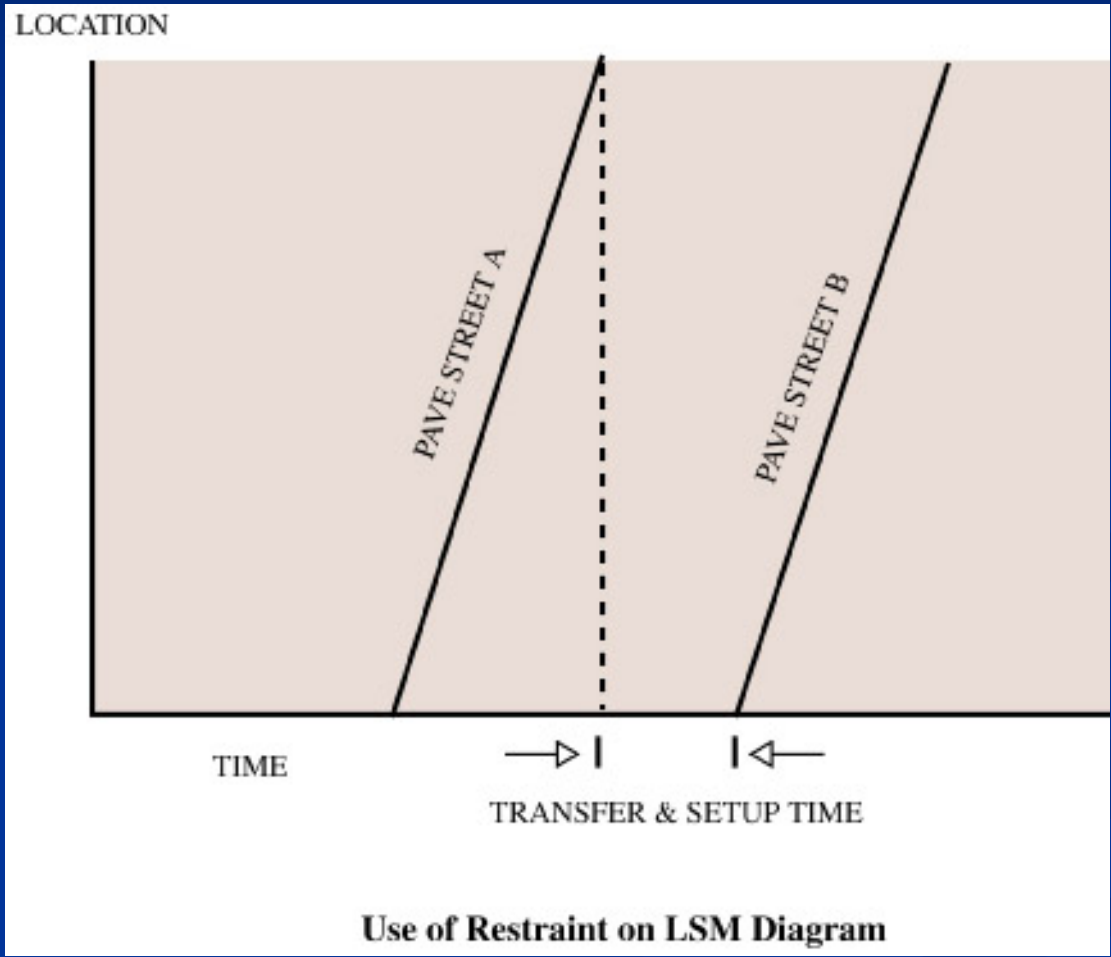



**Linear Scheduling Method Diagram**

# LOB Planning Steps

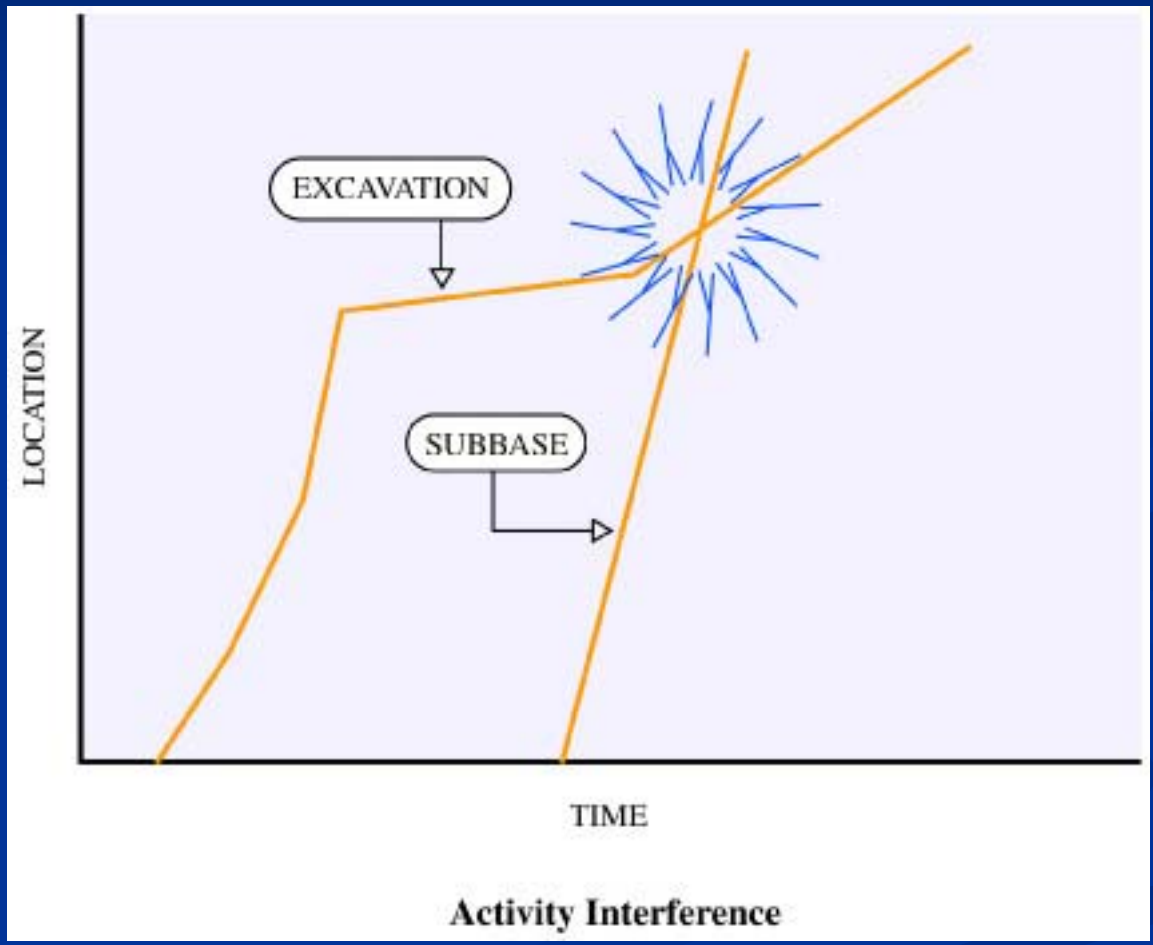
- Design Crews
- Determine Task for Crews
- Sequence of Trades
  - Location or Work Type
- Routing around the Job
- Buffer between Trade Crews

# Use of Restraint on LSM Diagram



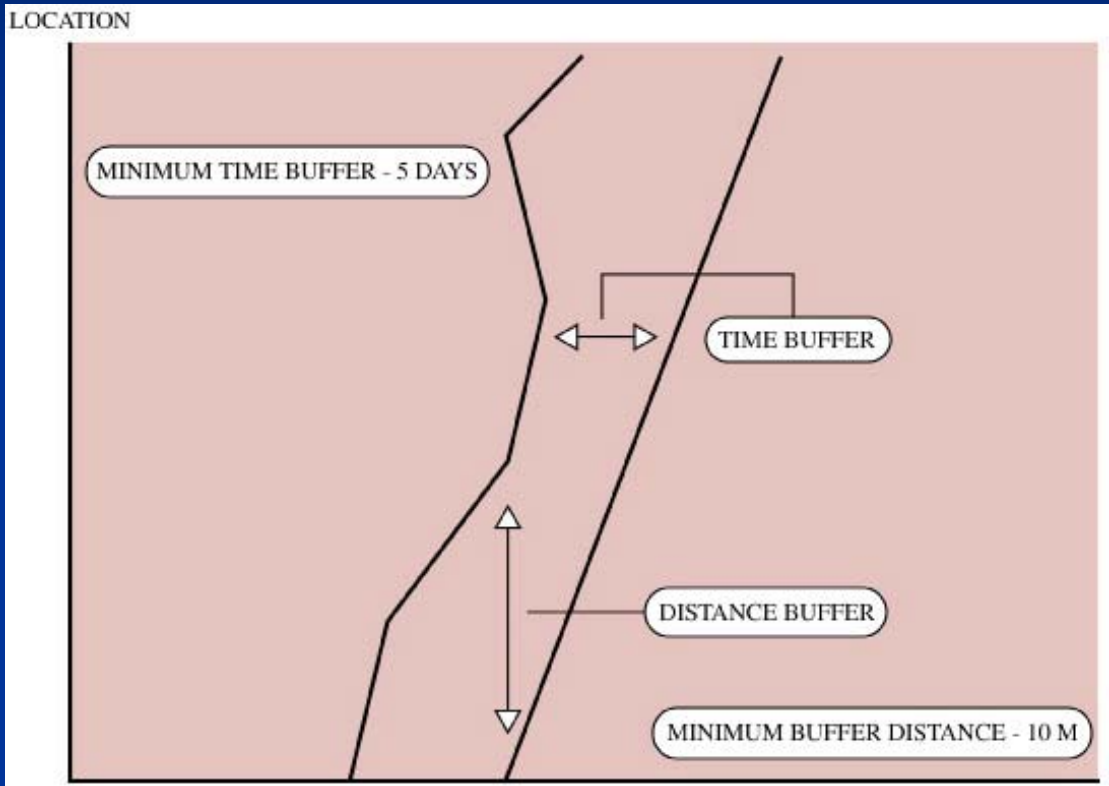

Use of Restraint on LSM Diagram

# Activity Interference



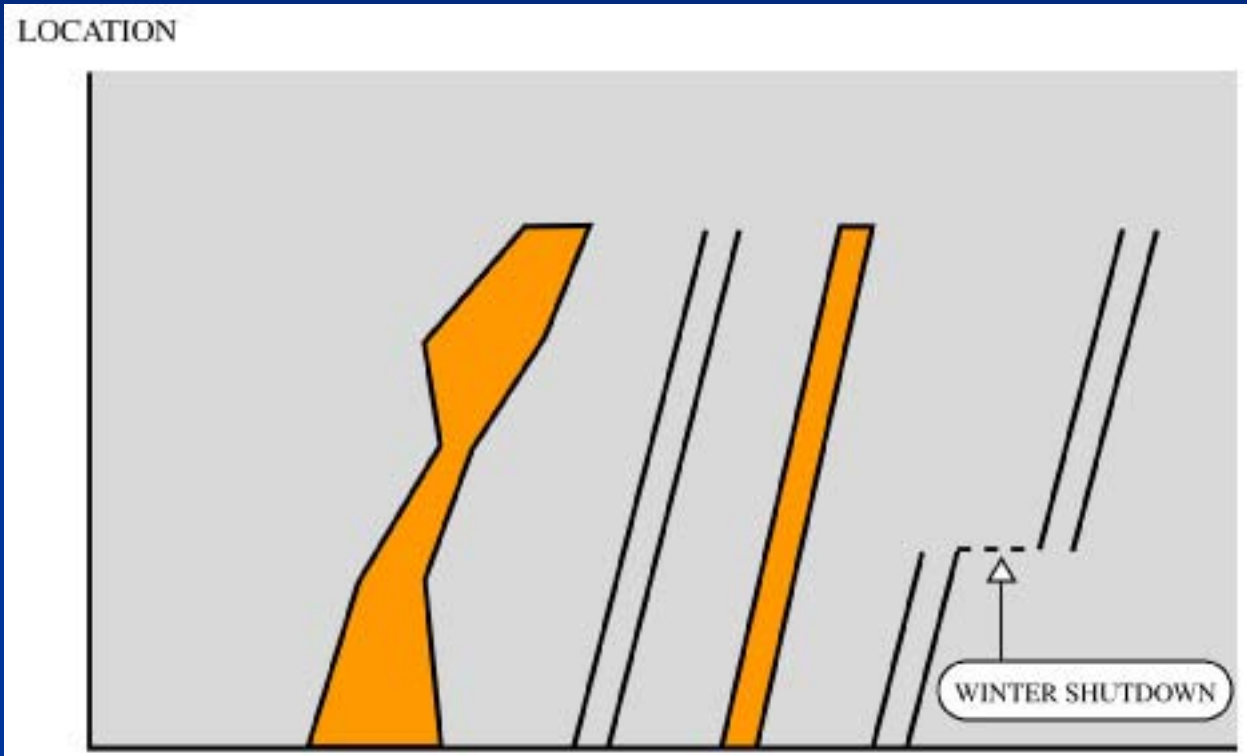

**Activity Interference**

# Use of Activity Buffers in LSM Schedules




**Use of Activity Buffers in LSM Schedules**

# Activity Intervals for LSM Schedules




**Activity Intervals for LSM Schedules**

# LOB Advantages

- Easier to Set-Up than Networks
- More Information Than Gantt Charts
- Graphically Shows
  - Rate of Progress
  - Activity Durations
  - Resource assignments
  - Resource assignments
  - Calendar dates
  - Graphical risk of conflicts
- Augment Network Scheduling on Projects with Repetitive and Discrete Activities
- Labor Histograms Can Be Prepared
- Resources May Be Balance
- Earnings Curves Can Be Developed

# LOB Disadvantages

- Less Effective When Work is Regularly Interrupted or When Activities Do Not Follow the Same Order in All Locations
- Difficult to Determine Minimum Interval and/or Buffers
- Hard to represent both horizontal and vertical progression