Scheduling Multiclass Queueing Networks via Fluid Models

John Hasenbein – OR/IE, UT-Austin
Ron Billings – OR/IE, UT-Austin
Leon Lasdon – MSIS, UT-Austin
Gideon Weiss – Statistics, Univ of Haifa

www.me.utexas.edu/~has

Support provided by NSF CAREER and NSF SBIR Grants

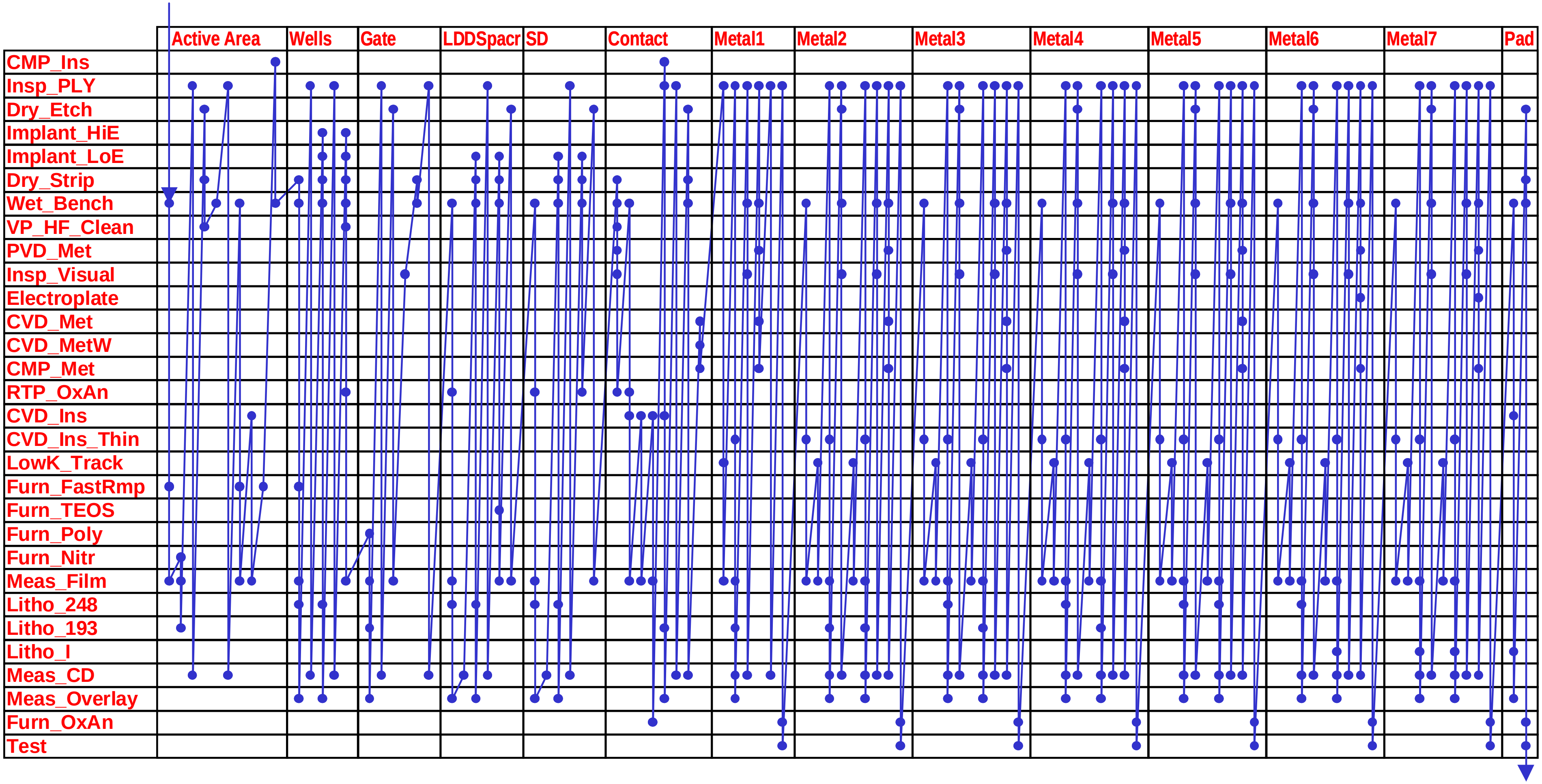
Tutorial Outline

- John – Motivating examples, background
- Ron – Solving fluid model problems, fluid model schedules, simulations
- Current literature – many theoretical results, some applications

Tutorial Outline

- Our focus
  - Develop a practical, scalable method for solving complex factory scheduling/dispatching problems
  - Provide an engine for solving general large scale fluid model problems
  - Provide a simulation tool for testing translation methods
  - Test the methodology on detailed, realistic wafer fab models

Typical Wafer Fab



Active Area, Wells, Gate, LDDSpacr, SD, Contact, Metal1, Metal2, Metal3, Metal4, Metal5, Metal6, Metal7, Pad

CMP_Ins
Insp_PLY
Dry_Etch
Implant_HiE
Implant_LoE
Dry_Strip
Wet_Bench
VP_HF_Clean
PVD_Met
Insp_Visual
Electroplate
CVD_Met
CVD_MetW
CMP_Met
RTP_OxAn
CVD_Ins
CVD_Ins_Thin
LowK_Track
Furn_FastRmp
Furn_TEOS
Furn_Poly
Furn_Nitr
Meas_Film
Litho_248
Litho_193
Litho_I
Meas_CD
Meas_Overlay
Furn_OxAn
Test

Stochastic Processing Networks - Features

- Network of workstations containing parallel tool groups
- Jobs move from station to station, different service type during subsequent visits
- Jobs divided into *classes* when waiting at a station
- Setups – machine setup required when switching processing tasks

Stochastic Processing Networks (SPN) – Features

- Batching – can form batches of jobs to be processed simultaneously
- Probabilistic/deterministic routing
- *Important:* how do we handle the transient nature of the system

Future Modelling Extensions

- Input control
- Assembly/disassembly
- Dynamic routing decisions

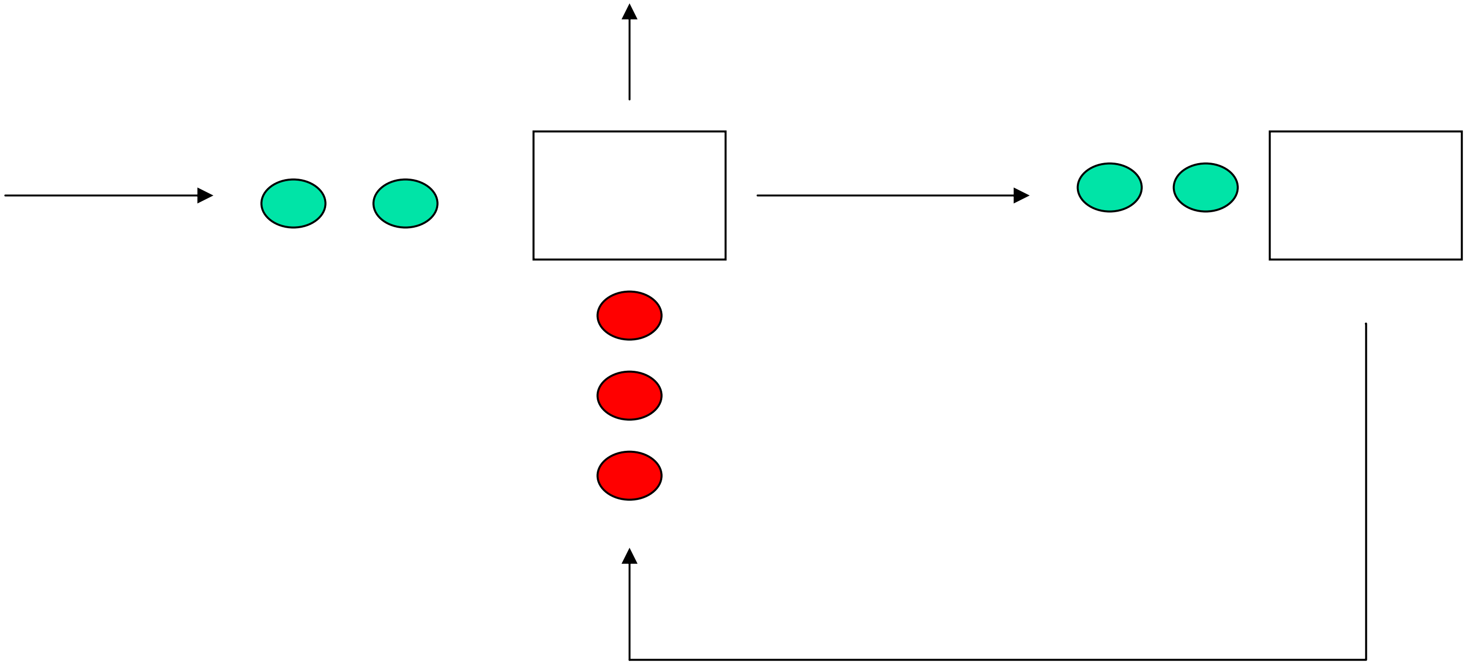
Goals and Performance Measures

- Minimize expected makespan – not useful for dynamic systems
- Minimize average WIP
- Minimize average cycle time
- Minimize average holding costs
  - Linear objective function
  - Convex objective function

Goals and Performance Measures

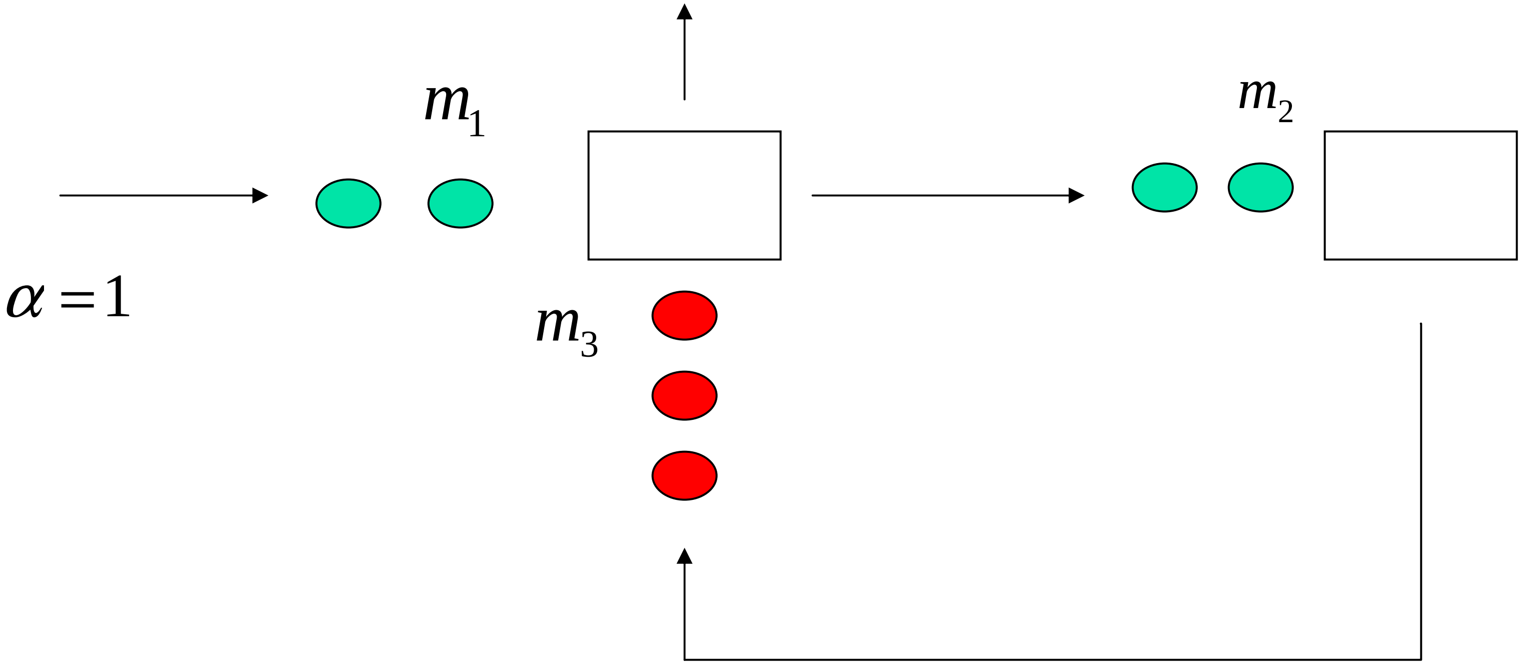
- Methods to produce “good” schedules for a general SPN
- Fast, implementable, adaptive

Scheduling Reentrant Systems



- Three class “reentrant line”
- Minimize long-run average WIP

Scheduling Reentrant Systems



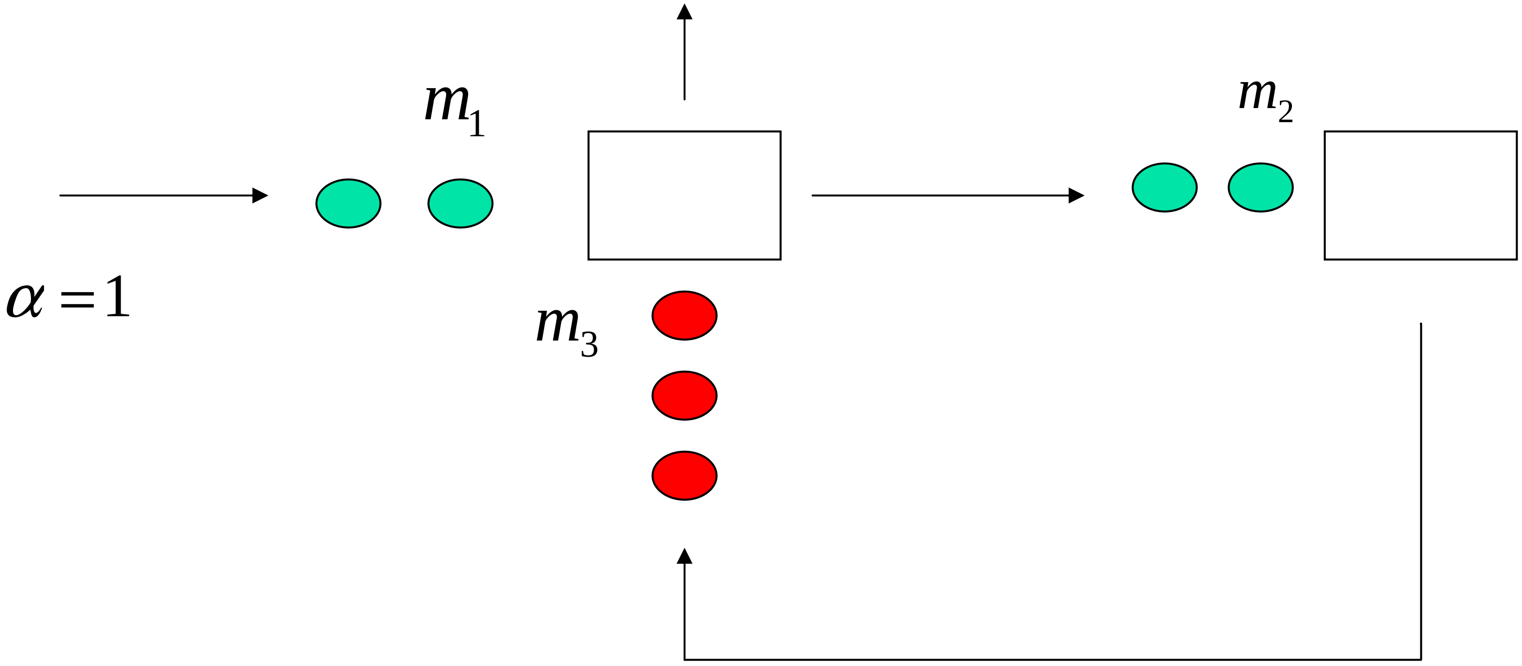
$m = (0.2, 0.9, 0.7)$

$Note \;\; m_1 + m_3 < 1 \;\; and \;\; m_2 < 1.$

Scheduling Observations

- For simplicity, suppose all service and interarrival times are exponential.
- Problem can be modeled as an infinite state Markov decision process (MDP).
- Can truncate state space and solve MDP.

Scheduling Reentrant Systems



$m = (0.2, 0.9, 0.7)$ *Note* $m_1 + m_3 < 1$ *and* $m_2 < 1.$

Performance of Scheduling Heuristics

Heuristic Idea	**Scheduling Policy**	**Avg WIP**
Greedy Draining	*Last Buffer FS*	28.94
MDP	*MDP*	20.68
Customer Fairness	*FIFO*	20.11
Pursue Target WIP	*Fluctuation Smoothing*	18.40
Starvation Avoidance	*First Buffer FS*	17.42

Performance of Scheduling Heuristics

Heuristic Idea	**Scheduling Policy**	**Avg WIP**
Greedy Draining	*Last Buffer FS*	28.94
MDP	*MDP*	20.68
Customer Fairness	*FIFO*	20.11
Pursue Target WIP	*Fluctuation Smoothing*	18.40
Starvation Avoidance	*First Buffer FS*	17.42
Balance LBFS/FBFS	*Threshold (5)*	16.68

Scheduling Results

- Theshold(5) policy – Avg WIP is 16.68
- Provable lower bounds on Avg WIP:
  - Bertsimas, et al.(polyhedral method): 9.88
  - JH (M/M/1 bound) : 11.58
  - MDP Bound: 12.18
- Optimal policy for this network is not known!

Fluid Model Approach

- Replace discrete jobs with fluid flow
- Replace workstation with “pumps”
- Fluid model is a continuous, deterministic approximation
- Fluid models are more tractable
- Require only mean value info

Fluid Models and Queueing Theory

- Some early applied papers in the 70’s and 80’s.
- Most theoretical advances achieved in the late 80’s and 90’s.
- Many papers in the last 2-3 years on fluid scheduling.
- Similar models in telecom literature.

The Fluid Bureaucracy

- **Fluid Flow Model of Networks of Queues**, James Vandergraft, *Management Science 1983*
- “A new technique is presented for modeling flow through a network of queues.”
- “An example of claims processing in a Social Security Administration’s District Office is given.”

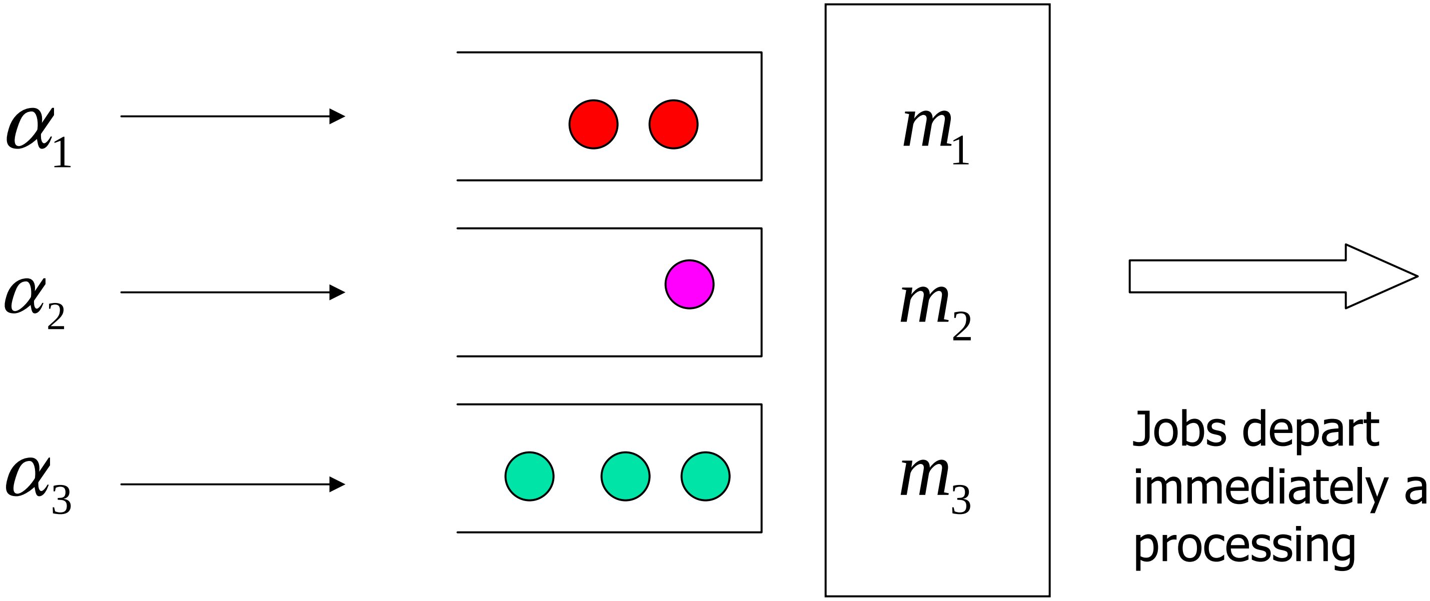
An Honorable Start in Garbage

- **Model for Optimal Operation and Design of Solid Waste Transfer Stations**, Harold Yaffe (UC-Berkeley), *Transportation Science, 1974.*
- “A deterministic queueing model is formulated … arrivals and departures of vehicles are treated as fluid flows.”
- “The model takes into account a time-dependent arrival rate of refuse ….”

Why Fluid Models Might Work, Part I - Stability Analysis

- Theorem (Dai 95): if the fluid model is stable, then any associated queueing network is stable (positive recurrent).
- Fluid model is “stable” if starting with any initial buffer levels, network will drain in finite time.
- Converse does not hold in general.

Why Fluid Models Might Work, Part II – Feedforward Station

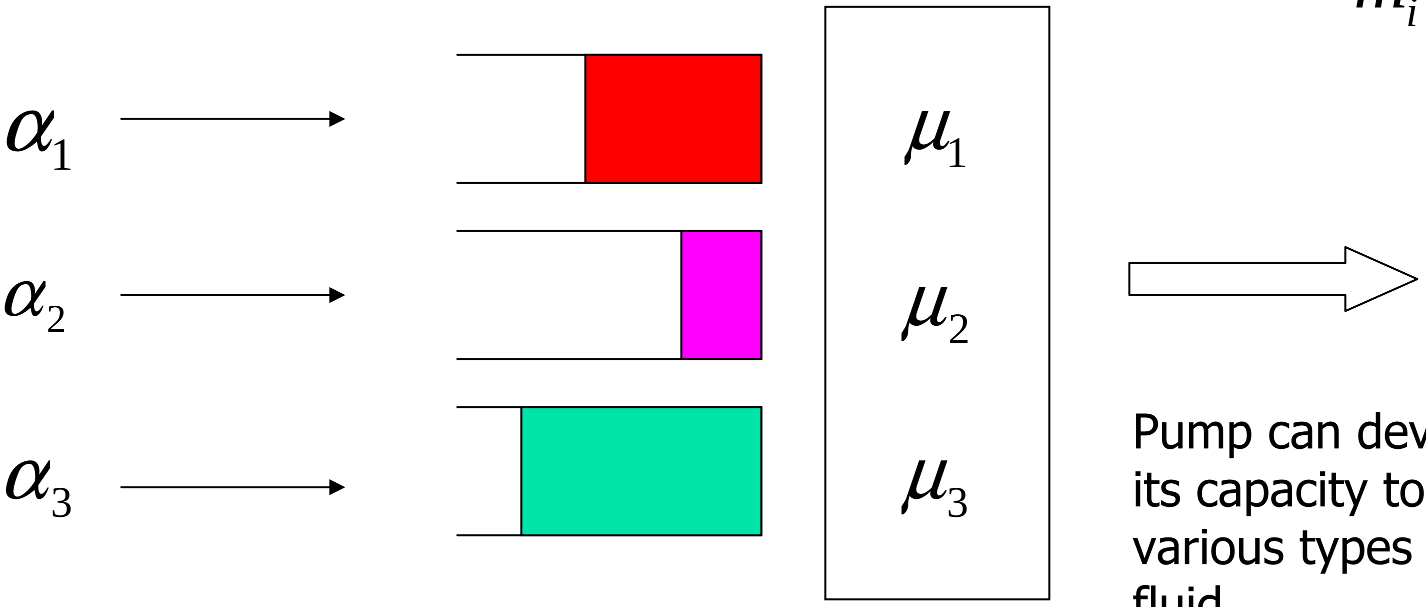


Jobs depart immediately after processing

Single server can process one job at a time

Single Station Case – The Fluid Model

$$\mu_i = \frac{1}{m_i}$$



Pump can devote its capacity to various types of fluid

Continuous fluid mass arrives from the outside

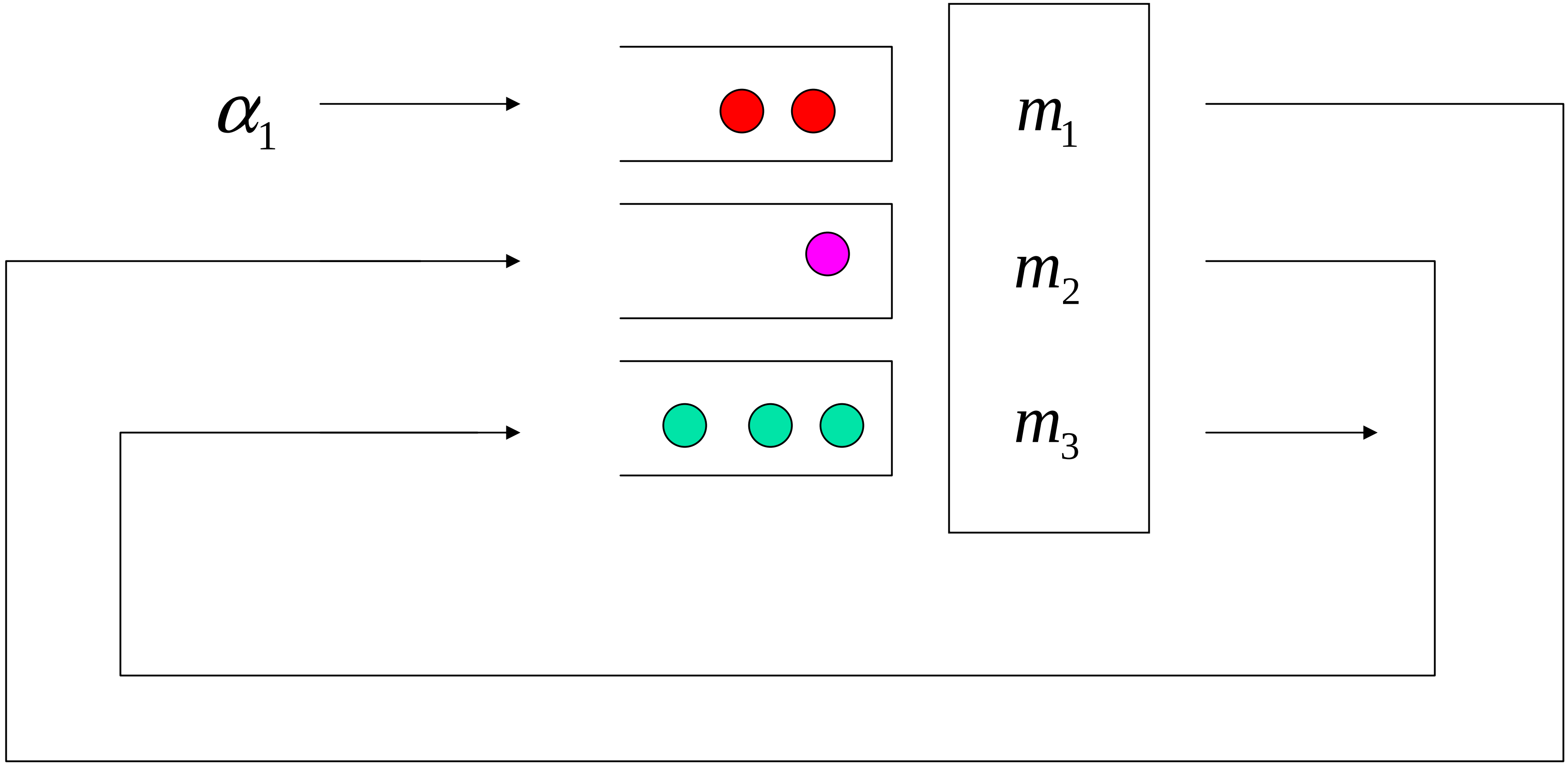
Optimal Fluid Model Solution Linear Holding Costs

- Suppose each type i fluid costs c_i dollars per unit time
- Optimal control (Avram, et al. 95):
  - prio$c_i$ritize fluid according to the “$c-\mu$ rule”
  - high priority fluid has absolute priority over lower priority fluid
  - processor sharing may occur
- This optimal control is *pathwise optimal*, it minimizes the cost at all time points.

Optimal Scheduling Rule – Single Station Queueing Model

- Cox and Smith 1961
- $c-\mu$ rule minimizes long-run average holding costs over all non-preemptive scheduling rules.
- Assumptions
  - Arbitrary service distributions
  - Poisson arrivals

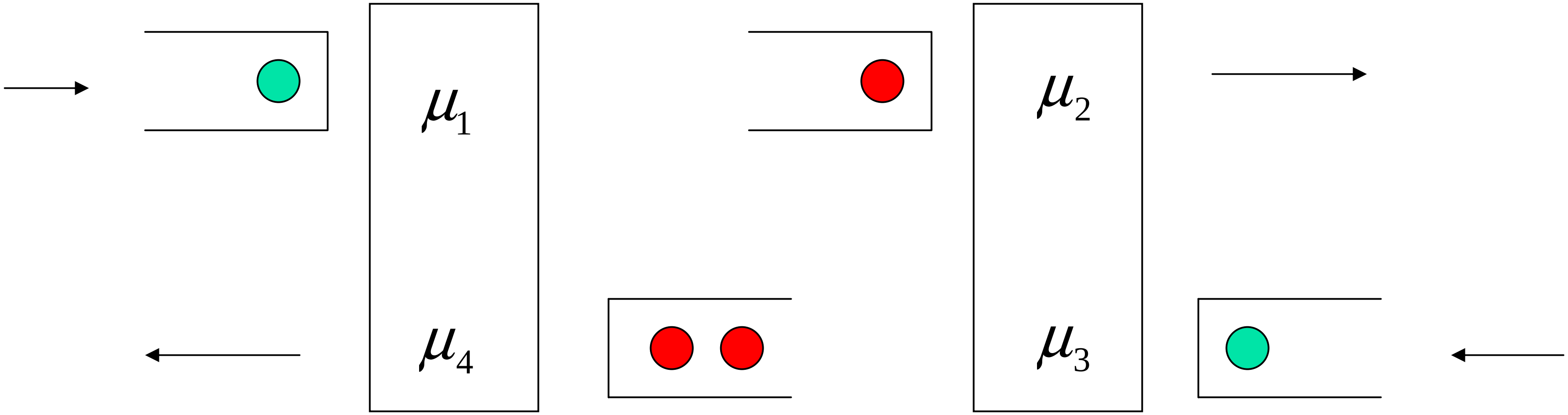
Why Fluid Models Might Work, Part III – Reentrant Lines



Optimal Policies for Fluid and Queueing Networks

- Klimov’s rule is pathwise optimal for the fluid model (Chen and Yao 93, Weiss 95).
- “Klimov’s Model” 1974.
- Klimov’s rule is optimal for the queueing model among all non-preemptive policies.
- Results hold for single-station multiclass fluid and queueing networks with arbitrary routing.

Not as easy as it looks - Kumar-Seidman Network



Exit classes are the slow classes

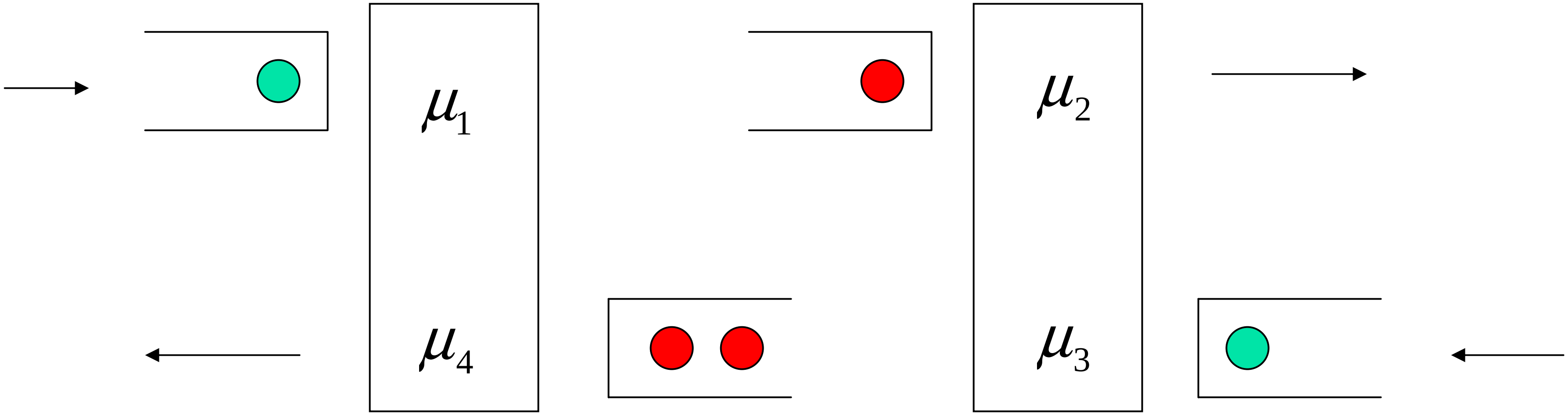
$$\mu_1 = \mu_3 = 6 \quad and \quad \mu_2 = \mu_4 = 1.5$$

Kumar-Seidman Network Optimal Fluid Policy

- Last-Buffer-First Served is Optimal
- Give fluids 2 and 4 high priority
- If buffers 2 and 4 are empty, buffers 1 and 3 may be worked on
- Split server effort to keep 2 and 4 empty

Naïve translation of fluid policy

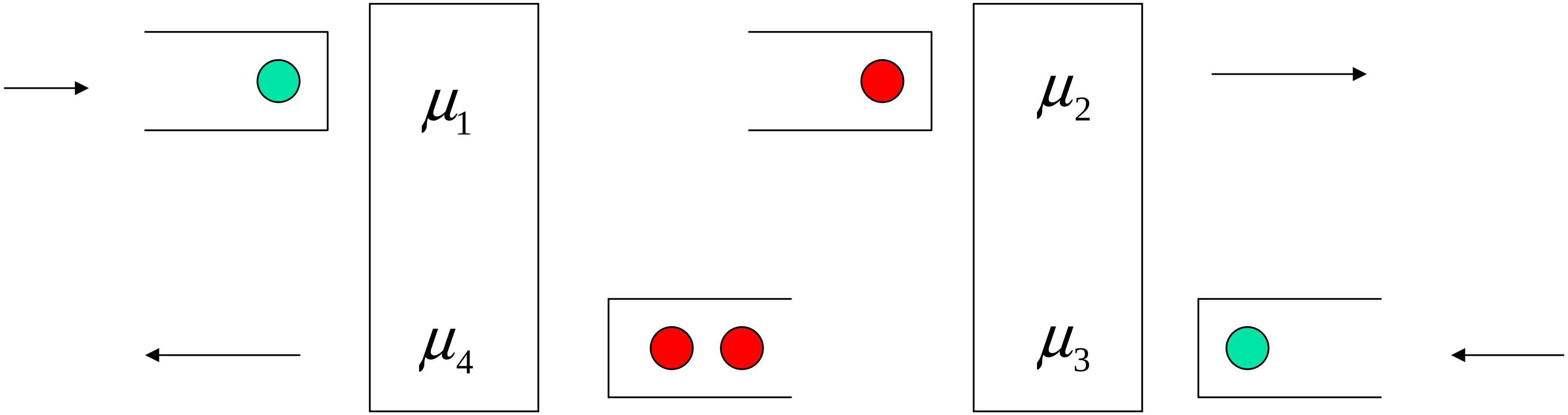
Give exit classes high priority.



This policy is unstable in the queueing model – queue lengths will explode with probability 1!

Less Naïve translation of fluid policy

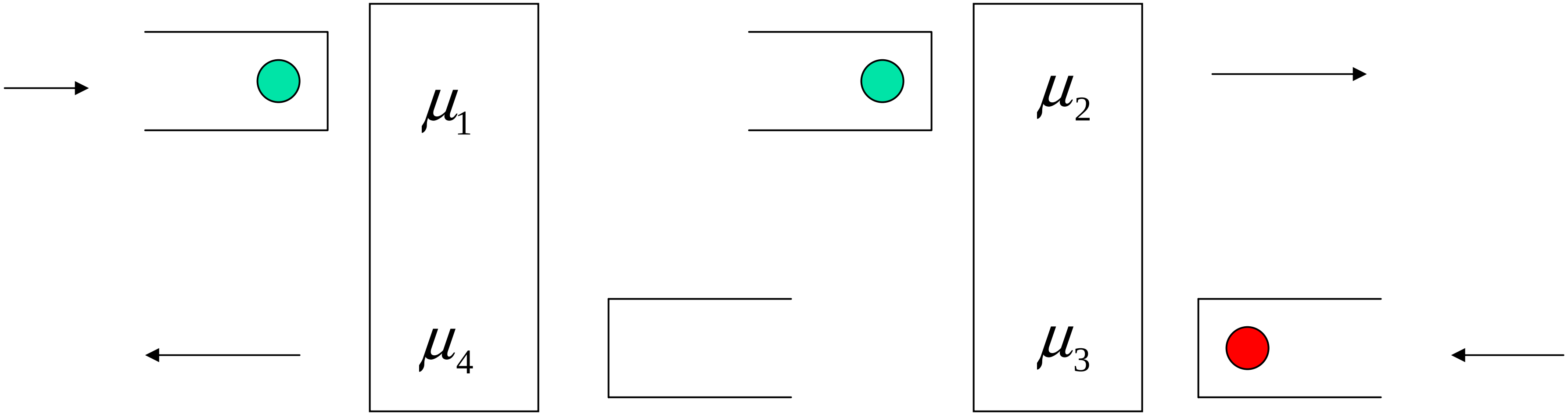
LBFS with priority reversal.



Reverse priorities when an exit buffer becomes empty – avoid starvation.

Less Naïve translation of fluid policy

LBFS with priority reversal.



This policy is stable but still performs poorly.
Depending on parameters, need to switch sooner!

Translation Methods – A Sampling

- Bertsimas, Sethuraman, "*From fluid relaxations to practical algorithms for job shop scheduling: the makespan objective*," Math Programming, Series A, 2002.
- Meyn, "*Sequencing and Routing in Multiclass Queueing Networks I: Feedback Regulation*," SIAM Journal on Control and Optimization, 2001.

A Sampling of Recent Work

- Veatch, "*Using Fluid Solutions in Dynamic Scheduling*," To appear, Proc. of the 2001 Tinos Workshop on Manufacturing Systems.
- Dai and Weiss, "*A fluid heuristic for minimizing makespan in job shops*," OR 2002.
- Maglaras, "Discrete-review policies for scheduling stochastic networks: trajectory tracking and fluid-scale asymptotic optimality," AAP 2000.

Fluid Model Constraints (all Linear)

- Conserve flow

$$\left(\mathbf{I}_K - \mathbf{P}^{\mathrm{T}}\right)\tilde{\mathbf{u}}(t) + \tilde{\mathbf{q}}(t) = \left(\mathbf{I}_K - \mathbf{P}^{\mathrm{T}}\right)\mathbf{u}(t_0) + \mathbf{q}(t_0) + \boldsymbol{\alpha}\left(t - t_0\right) \qquad \forall t \geq t_0$$

- Non-decreasing number of units processed

$$\tilde{\mathbf{u}}(t') - \tilde{\mathbf{u}}(t) \geq \mathbf{0}_{K,1} \qquad \forall t, t' : t_0 \leq t < t'$$

- Limited machine capacity

$$\mathbf{BD}(\mathbf{p})\left[\tilde{\mathbf{u}}(t') - \tilde{\mathbf{u}}(t)\right] \leq \mathbf{D}(\mathbf{a})\mathbf{m}(t' - t) \qquad \forall t, t' : t_0 \leq t < t'$$

- Non-negative queue lengths

$$\tilde{\mathbf{q}}(t) \geq \mathbf{0}_{K,1} \qquad \forall t \geq t_0$$

Objective Functions: Minimize Integral Over…

- Makespan

$$\min_{\tilde{\mathbf{q}},\tilde{\mathbf{u}}} \tilde{C}_{\mathrm{max}}(\tilde{\mathbf{q}},\tilde{\mathbf{u}} \mid T) = \int_{t_0}^{t_0+T} \operatorname{sgn}\left(\left\|\tilde{\mathbf{q}}(t)\right\|_1\right) dt$$

- Weighted Holding Cost

$$\min_{\tilde{\mathbf{q}},\tilde{\mathbf{u}}} \tilde{C}_{\mathrm{h}}(\tilde{\mathbf{q}},\tilde{\mathbf{u}} \mid T) = \int_{t_0}^{t_0+T} \mathbf{c}^{\mathrm{T}}\tilde{\mathbf{q}}(t)\, dt$$

- Maximum Workload

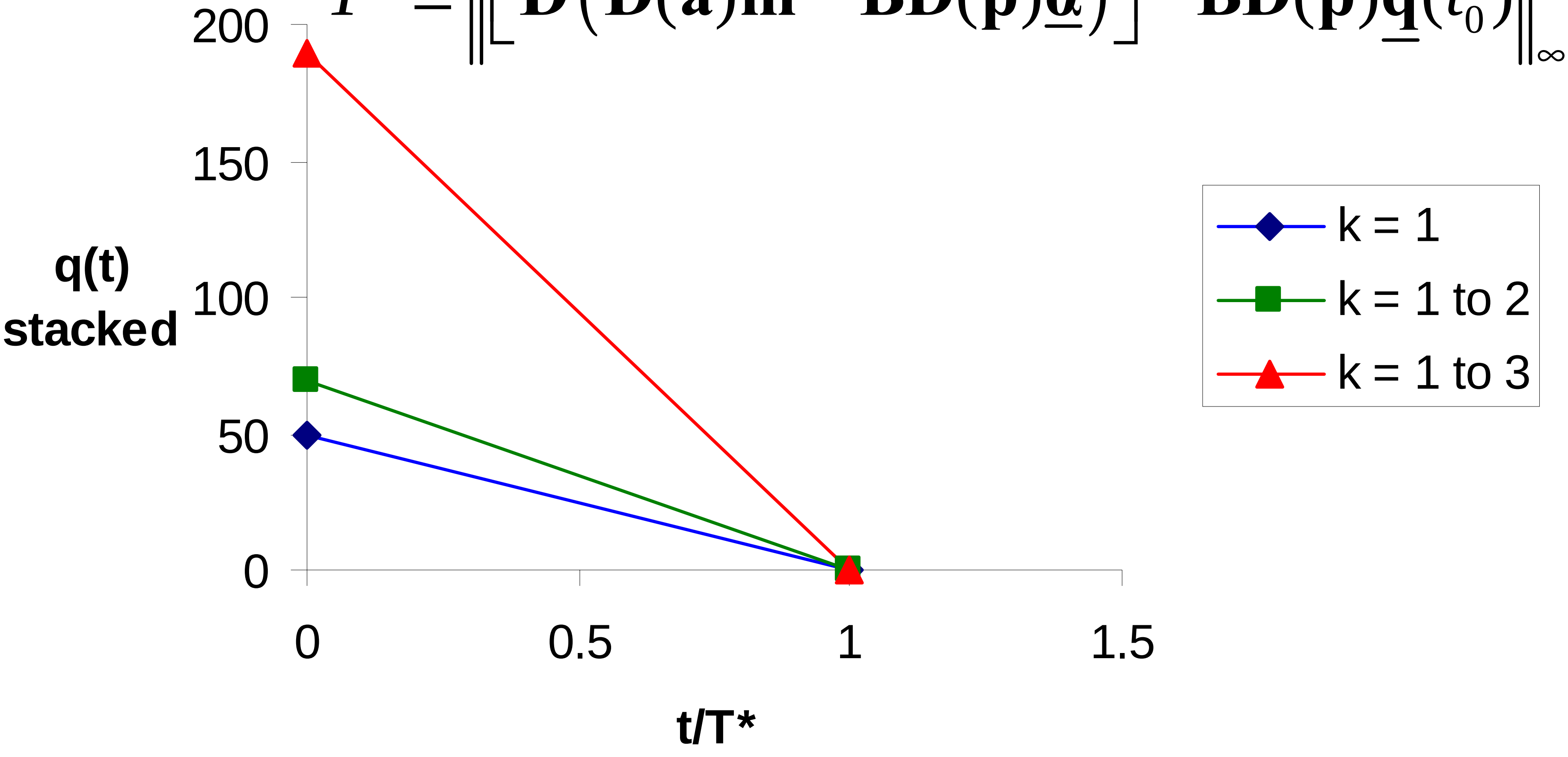
$$\min_{\tilde{\mathbf{q}},\tilde{\mathbf{u}}} \tilde{C}_{\mathrm{w}}(\tilde{\mathbf{q}},\tilde{\mathbf{u}} \mid T) = \int_{t_0}^{t_0+T} \left\|\tilde{\mathbf{w}}(t)\right\|_\infty dt$$

- Combination of Last Two

$$\tilde{\mathbf{w}}(t) \equiv \left[\mathbf{D}(\mathbf{a})\mathbf{D}(\mathbf{m})\right]^{-1}\mathbf{B}\mathbf{D}(\mathbf{p})\tilde{\mathbf{q}}(t)$$

An Optimal Makespan Solution

$$T^* \equiv \left\| \left[\mathbf{D}\left(\mathbf{D}(\mathbf{a})\mathbf{m} - \mathbf{BD}(\mathbf{p})\underline{\boldsymbol{\alpha}}\right)\right]^{-1} \mathbf{BD}(\mathbf{p})\underline{\mathbf{q}}(t_0) \right\|_\infty$$



Continuous Linear Program (CLP): Weighted Holding Cost

$$\min_{\tilde{\mathbf{q}},\tilde{\mathbf{u}}} \tilde{C}_{\mathrm{h}}(\tilde{\mathbf{q}},\tilde{\mathbf{u}} \mid T) = \int_{t_0}^{t_0+T} \mathbf{c}^{\mathrm{T}}\tilde{\mathbf{q}}(t)dt$$

$$\begin{array}{lll}
 & \left(\mathbf{I}_K - \mathbf{P}^{\mathrm{T}}\right)\tilde{\mathbf{u}}(t) + \tilde{\mathbf{q}}(t) & \\
\text{s.t.} & \quad = \left(\mathbf{I}_K - \mathbf{P}^{\mathrm{T}}\right)\mathbf{u}(t_0) & \forall t \geq t_0 \\
 & \quad + \mathbf{q}(t_0) + \boldsymbol{\alpha}\left(t - t_0\right) & \\
 & \tilde{\mathbf{u}}(t') - \tilde{\mathbf{u}}(t) \geq \mathbf{0}_{K,1} & \forall t, t' : t_0 \leq t < t' \\
 & \mathbf{BD}(\mathbf{p})\left[\tilde{\mathbf{u}}(t') - \tilde{\mathbf{u}}(t)\right] & \\
 & \quad \leq \mathbf{D}(\mathbf{a})\mathbf{m}(t' - t) & \forall t, t' : t_0 \leq t < t' \\
 & \tilde{\mathbf{q}}(t) \geq \mathbf{0}_{K,1} & \forall t \geq t_0
\end{array}$$

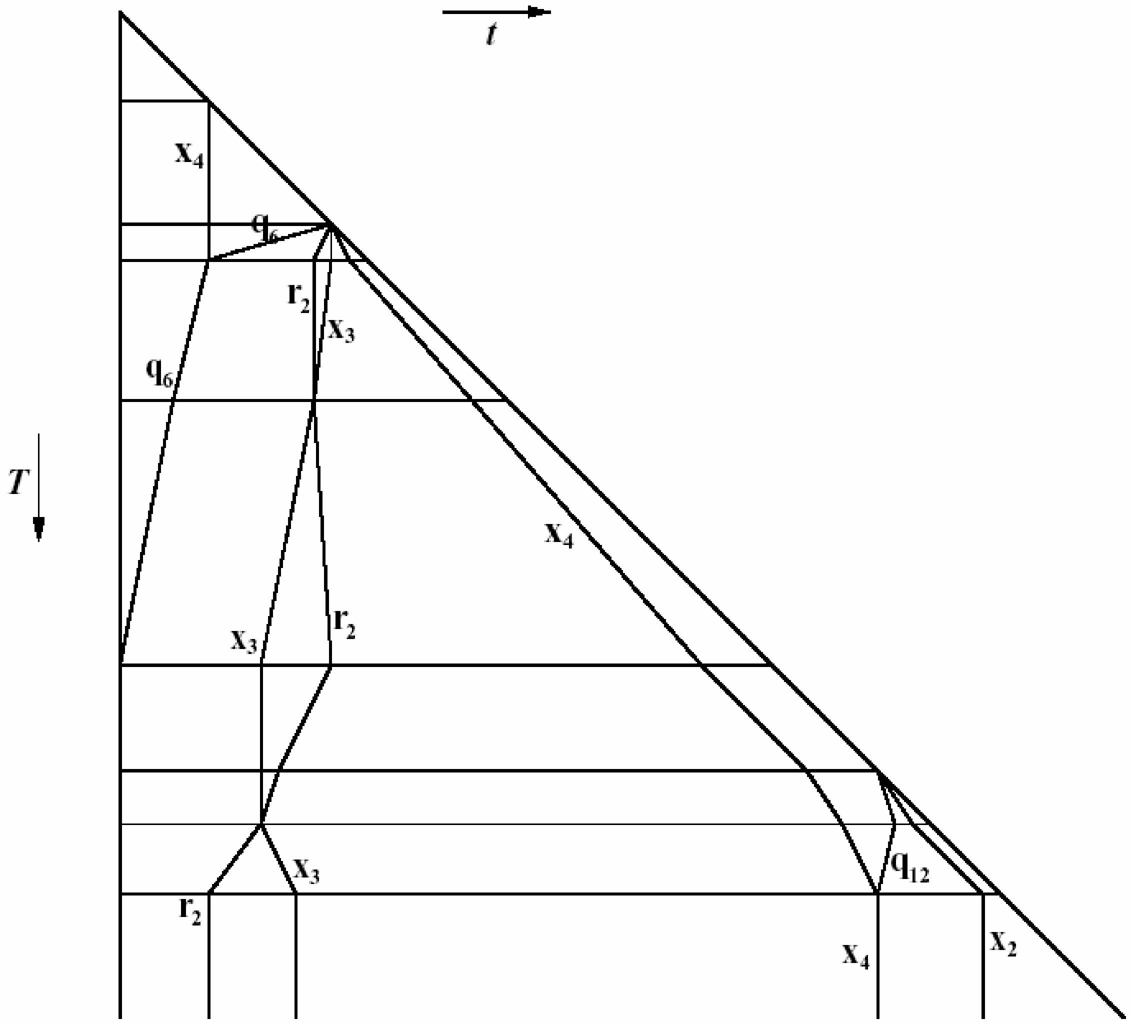
Separated Continuous Linear Program (SCLP)

$$\max_{\dot{\tilde{\mathbf{u}}}} \tilde{C}'_{\mathrm{h}}(\dot{\tilde{\mathbf{u}}}, \tilde{\mathbf{q}}, \dot{\tilde{\mathbf{y}}} \mid T) = \int_{t_0}^{t_0+T} (t_0 + T - t)\underline{\mathbf{c}}^{\mathrm{T}} \dot{\tilde{\mathbf{u}}}(t)dt$$

$$\begin{aligned}
\text{s.t.} \quad & \int_{t_0}^{t} \left(\mathbf{I}_K - \mathbf{P}^{\mathrm{T}}\right) \dot{\tilde{\mathbf{u}}}(s)ds + \tilde{\mathbf{q}}(t) = \mathbf{q}(t_0) + \boldsymbol{\alpha}\left(t - t_0\right) && \forall t \ge t_0 \\
& \mathbf{BD}(\mathbf{p})\dot{\tilde{\mathbf{u}}}(t) + \dot{\tilde{\mathbf{y}}}(t) = \mathbf{D}(\mathbf{a})\mathbf{m} && \forall t \ge t_0 \\
& \dot{\tilde{\mathbf{u}}}(t) \ge \mathbf{0}_{K,1} && \forall t \ge t_0 \\
& \tilde{\mathbf{q}}(t) \ge \mathbf{0}_{K,1} && \forall t \ge t_0 \\
& \dot{\tilde{\mathbf{y}}}(t) \ge \mathbf{0}_{I,1} && \forall t \ge t_0
\end{aligned}$$

Pullan (‘95): optimal piecewise constant solution

Gideon Weiss' SCLP Solution Method

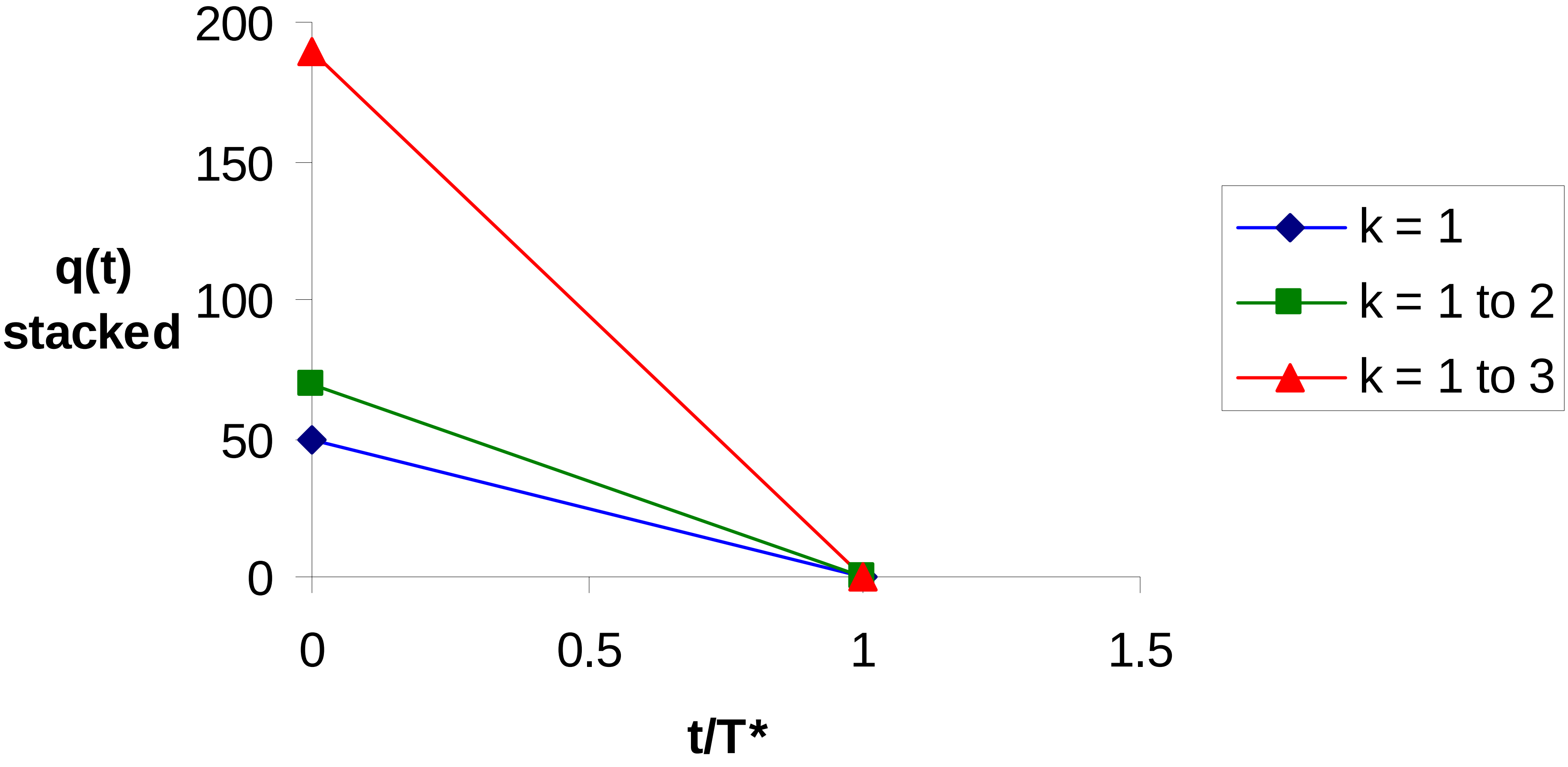


Non-Convex (Bilinear) Quadratic Program (QP)

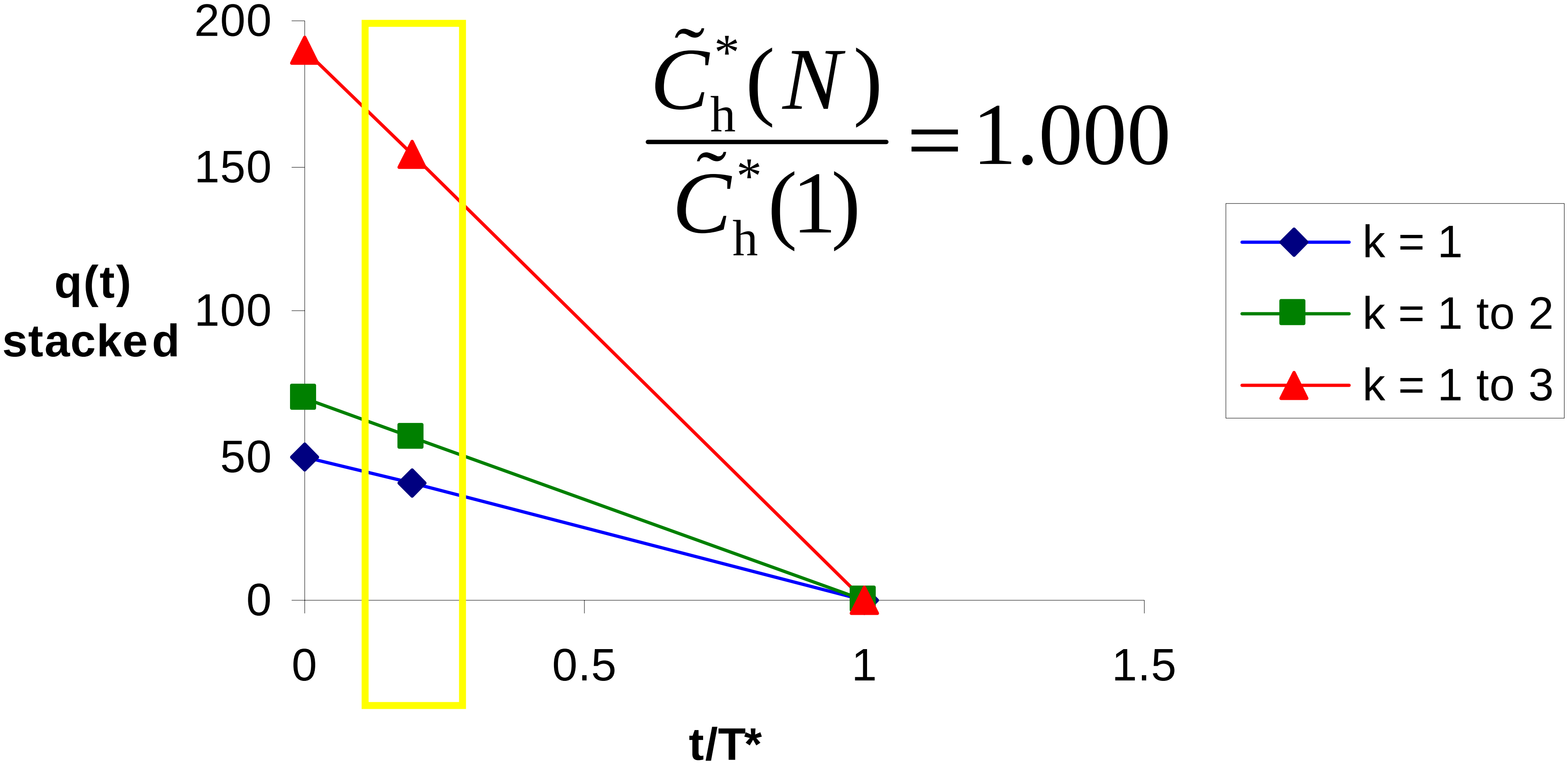
$$\min_{\Delta\mathbf{t},\tilde{\mathbf{Q}},\Delta\tilde{\mathbf{U}}} \tilde{C}_{\mathrm{h}}(\Delta\mathbf{t},\tilde{\mathbf{Q}},\Delta\tilde{\mathbf{U}},\Delta\tilde{\mathbf{Y}} \mid N) = \frac{\mathbf{c}^{\mathrm{T}}}{2}\left[\mathbf{q}(t_0)\Delta t_1 + \sum_{n=1}^{N-1}\tilde{\mathbf{q}}(t_n)\left(\Delta t_n + \Delta t_{n+1}\right)\right]$$

$$\begin{aligned}
\text{s.t.}\quad & \left(\mathbf{I}_K - \mathbf{P}^{\mathrm{T}}\right)\Delta\tilde{\mathbf{u}}(t_1) + \tilde{\mathbf{q}}(t_1) - \boldsymbol{\alpha}\Delta t_1 = \mathbf{q}(t_0) & & \\
& \left(\mathbf{I}_K - \mathbf{P}^{\mathrm{T}}\right)\Delta\tilde{\mathbf{u}}(t_n) + \tilde{\mathbf{q}}(t_n) - \tilde{\mathbf{q}}(t_{n-1}) - \boldsymbol{\alpha}\Delta t_n = \mathbf{0}_{K,1} & & \forall n \in \{2,3,\dots,N-1\} \\
& \left(\mathbf{I}_K - \mathbf{P}^{\mathrm{T}}\right)\Delta\tilde{\mathbf{u}}(t_N) - \tilde{\mathbf{q}}(t_{N-1}) - \boldsymbol{\alpha}\Delta t_N = \mathbf{0}_{K,1} & & \\
& \mathbf{BD}(\mathbf{p})\Delta\tilde{\mathbf{u}}(t_n) - \mathbf{D}(\mathbf{a})\mathbf{m}\Delta t_n \le \mathbf{0}_{I,1} & & \forall n \in \{1,2,\dots,N\} \\
& \Delta t_n \ge 0 & & \forall n \in \{1,2,\dots,N\} \\
& \tilde{\mathbf{q}}(t_n) \ge \mathbf{0}_{K,1} & & \forall n \in \{1,2,\dots,N-1\} \\
& \Delta\tilde{\mathbf{u}}(t_n) \ge \mathbf{0}_{K,1} & & \forall n \in \{1,2,\dots,N\}
\end{aligned}$$

Optimal QP Solution for *N* = 1



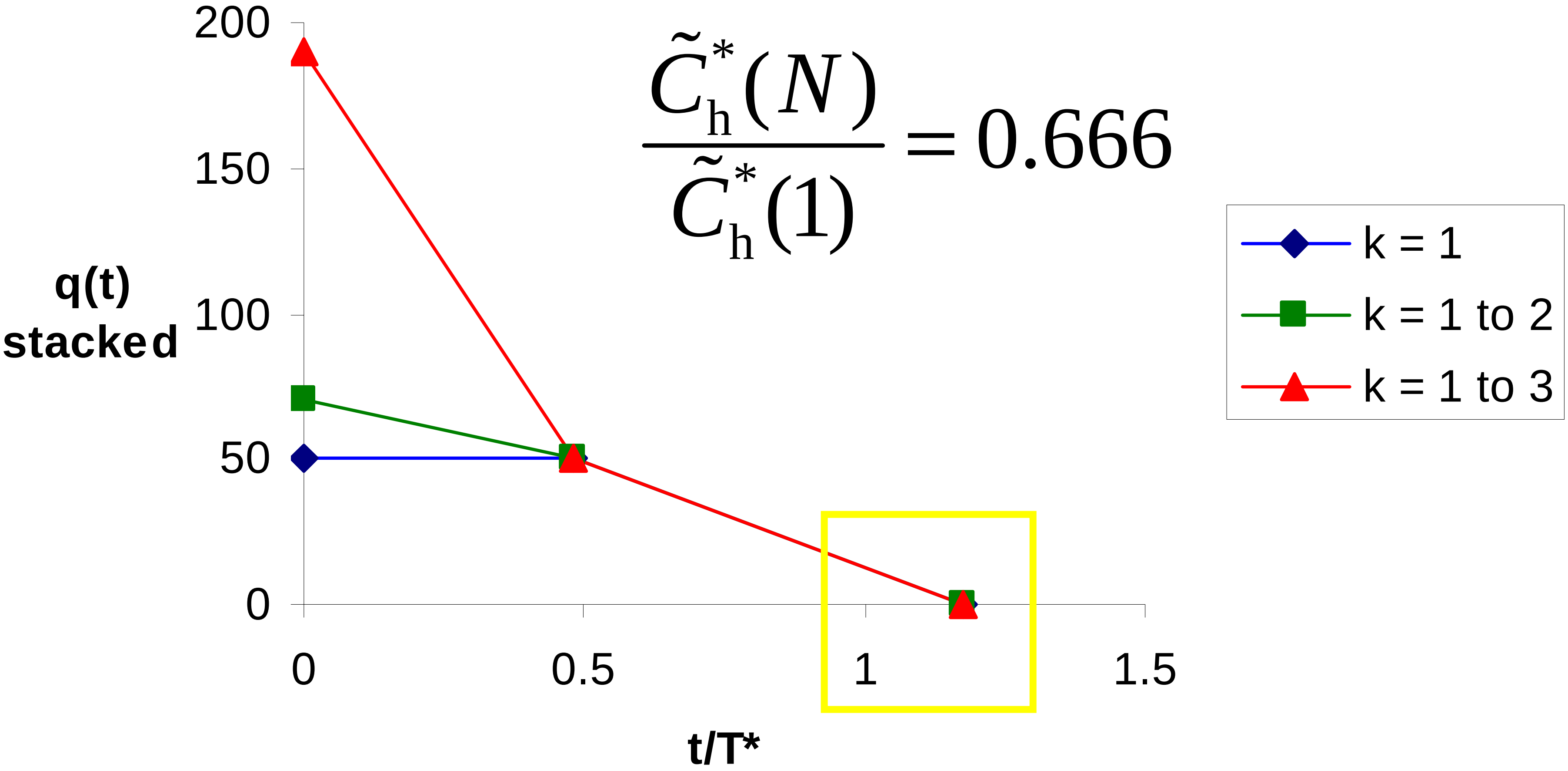
Feasible QP Solution for *N* = 2



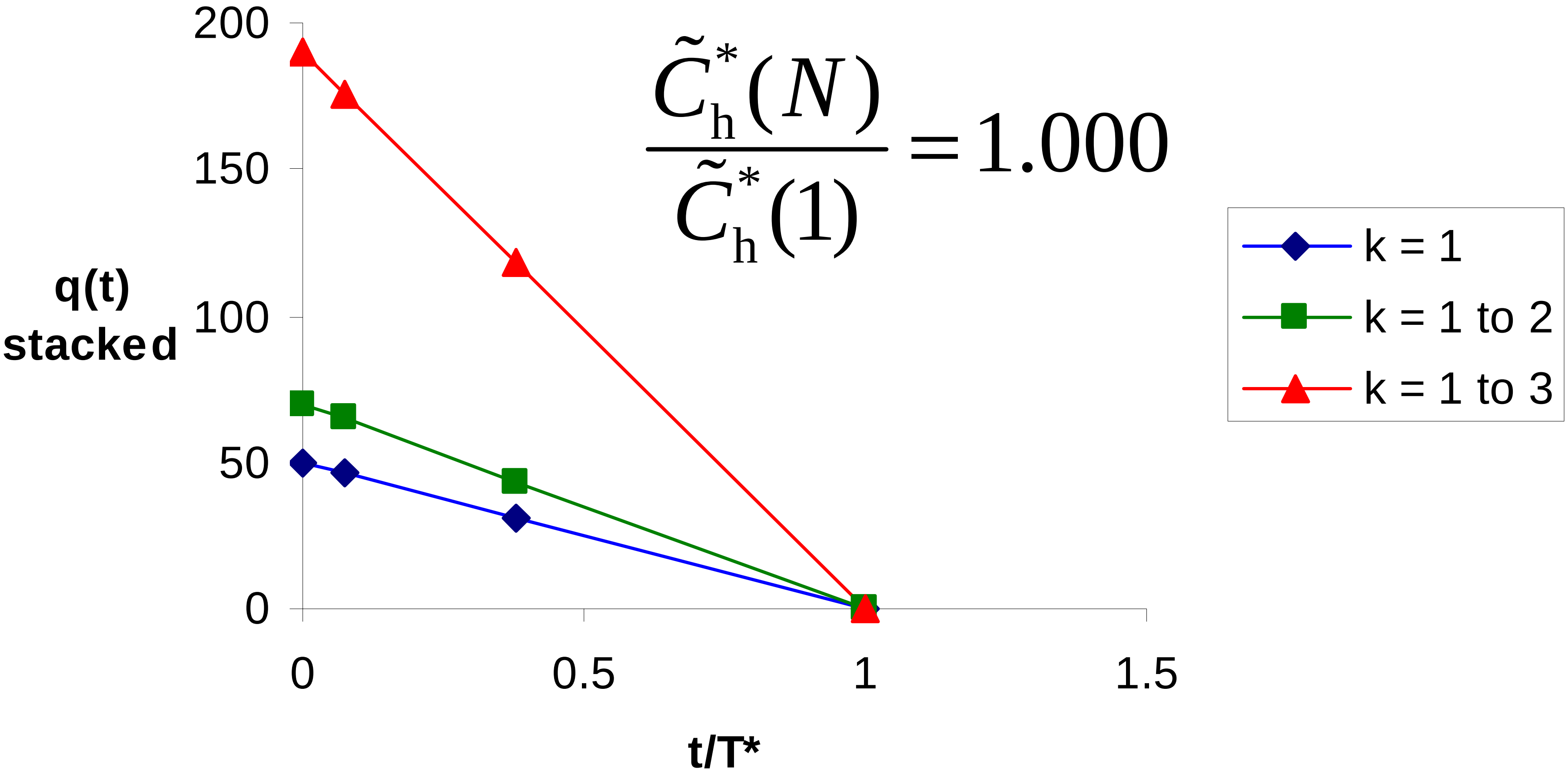
$$\frac{\tilde{C}_{\mathrm{h}}^{*}(N)}{\tilde{C}_{\mathrm{h}}^{*}(1)} = 1.000$$

Optimal QP Solution for *N* = 2

$$\frac{\tilde{C}_{\mathrm{h}}^{*}(N)}{\tilde{C}_{\mathrm{h}}^{*}(1)} = 0.666$$

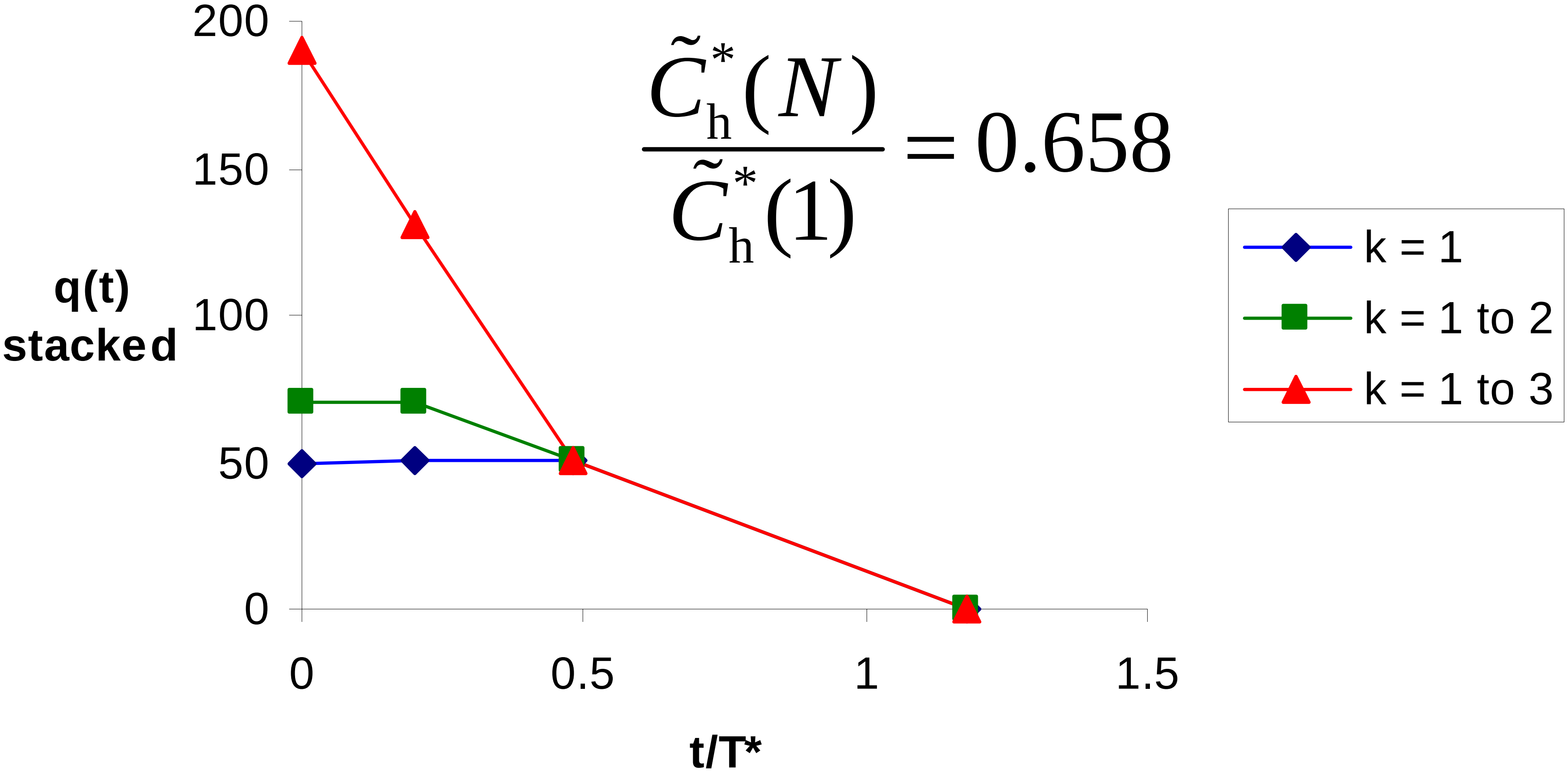


Feasible QP Solution for *N* = 3



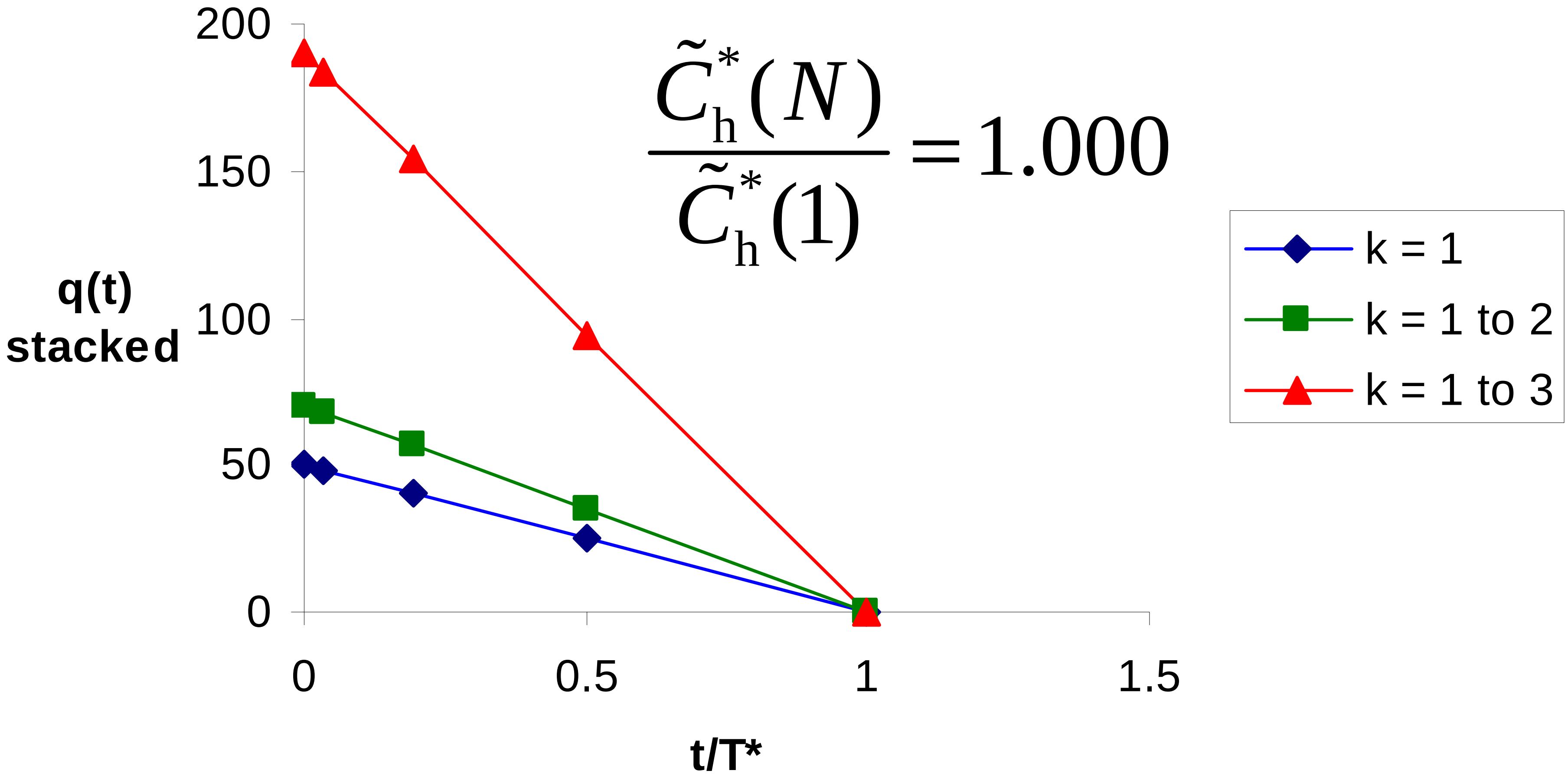
$$\frac{\tilde{C}_{\mathrm{h}}^{*}(N)}{\tilde{C}_{\mathrm{h}}^{*}(1)} = 1.000$$

Optimal QP Solution for *N* = 3



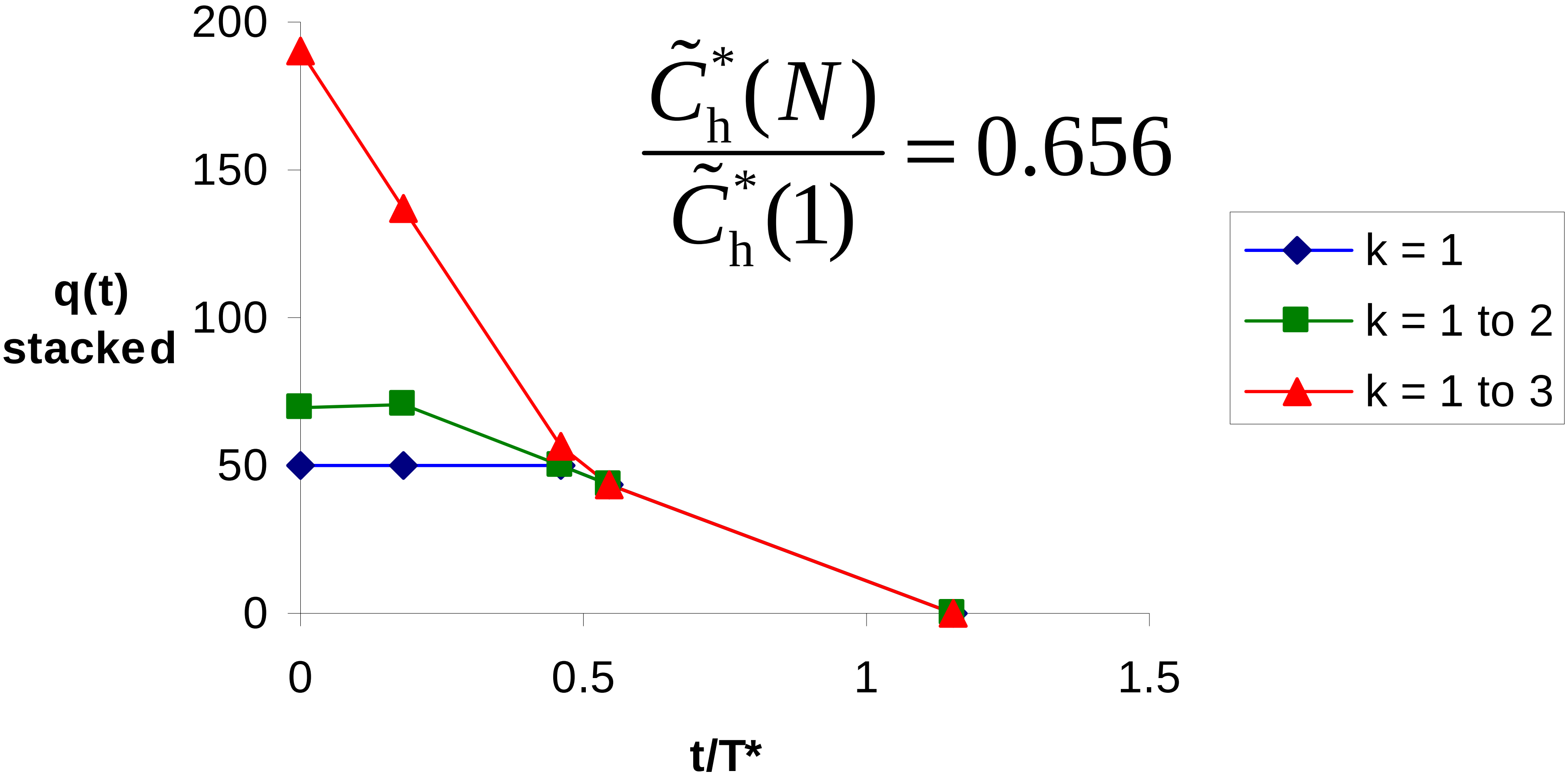
$$\frac{\tilde{C}_{\mathrm{h}}^{*}(N)}{\tilde{C}_{\mathrm{h}}^{*}(1)} = 0.658$$

Feasible QP Solution for *N* = 4



$$\frac{\tilde{C}_{\mathrm{h}}^{*}(N)}{\tilde{C}_{\mathrm{h}}^{*}(1)} = 1.000$$

Optimal QP Solution for $N \geq 4$



$$\frac{\tilde{C}^*_{\mathrm{h}}(N)}{\tilde{C}^*_{\mathrm{h}}(1)} = 0.656$$

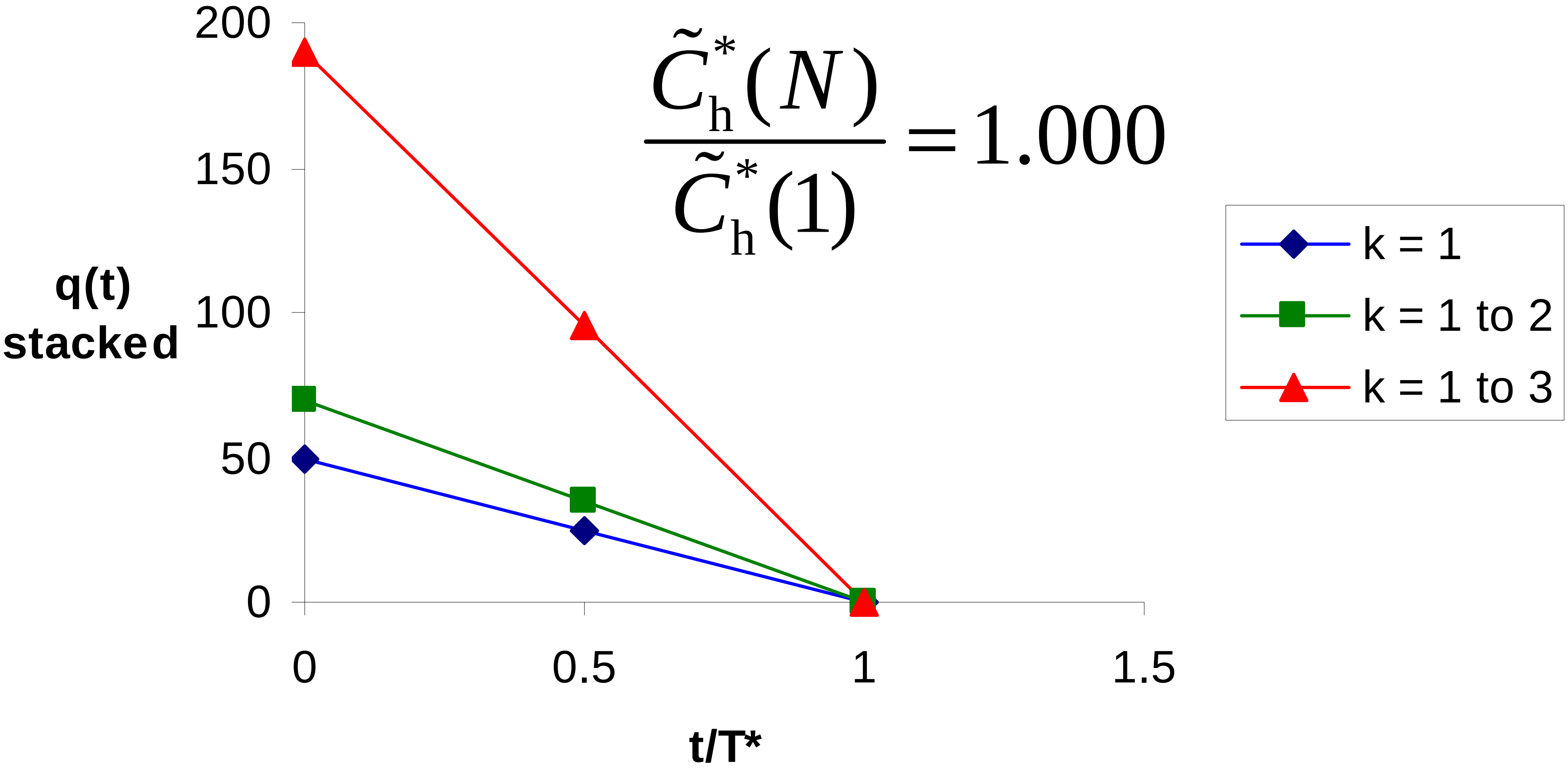
Linear Program (LP) with Fixed Time Intervals

$$\min_{\underline{\tilde{\mathbf{Q}}}} \tilde{C}_{\mathrm{h}}\left(\underline{\tilde{\mathbf{Q}}}, \Delta\tilde{\mathbf{U}}, \Delta\tilde{\mathbf{Y}}, \tilde{\mathbf{Q}} \mid N, \Delta\mathbf{t}\right) = \frac{\underline{\mathbf{c}}^{\mathrm{T}}}{2}\left[\underline{\mathbf{q}}(t_0)\Delta t_1 + \sum_{n=1}^{N-1}\underline{\tilde{\mathbf{q}}}(t_n)\left(\Delta t_n + \Delta t_{n+1}\right)\right]$$

$$\begin{aligned}
\text{s.t.}\quad & \underline{\tilde{\mathbf{q}}}(t_1) \le \underline{\mathbf{q}}(t_0) + \underline{\boldsymbol{\alpha}}\Delta t_1 & \\
& -\underline{\tilde{\mathbf{q}}}(t_{n-1}) + \underline{\tilde{\mathbf{q}}}(t_n) \le \underline{\boldsymbol{\alpha}}\Delta t_n & \forall n \in \{2, 3, \ldots, N-1\} \\
& -\underline{\tilde{\mathbf{q}}}(t_{N-1}) \le \underline{\boldsymbol{\alpha}}\Delta t_N & \\
& -\mathbf{BD}(\mathbf{p})\underline{\tilde{\mathbf{q}}}(t_1) \le -\mathbf{BD}(\mathbf{p})\underline{\mathbf{q}}(t_0) + \boldsymbol{\chi}\Delta t_1 & \\
& \mathbf{BD}(\mathbf{p})\left[\underline{\tilde{\mathbf{q}}}(t_{n-1}) - \underline{\tilde{\mathbf{q}}}(t_n)\right] \le \boldsymbol{\chi}\Delta t_n & \forall n \in \{2, 3, \ldots, N-1\} \\
& \mathbf{BD}(\mathbf{p})\underline{\tilde{\mathbf{q}}}(t_{N-1}) \le \boldsymbol{\chi}\Delta t_N & \\
& -\left(\mathbf{I}_K - \mathbf{P}^{\mathrm{T}}\right)\underline{\tilde{\mathbf{q}}}(t_n) \le \mathbf{0}_{K,1} & \forall n \in \{1, 2, \ldots, N-1\}
\end{aligned}$$

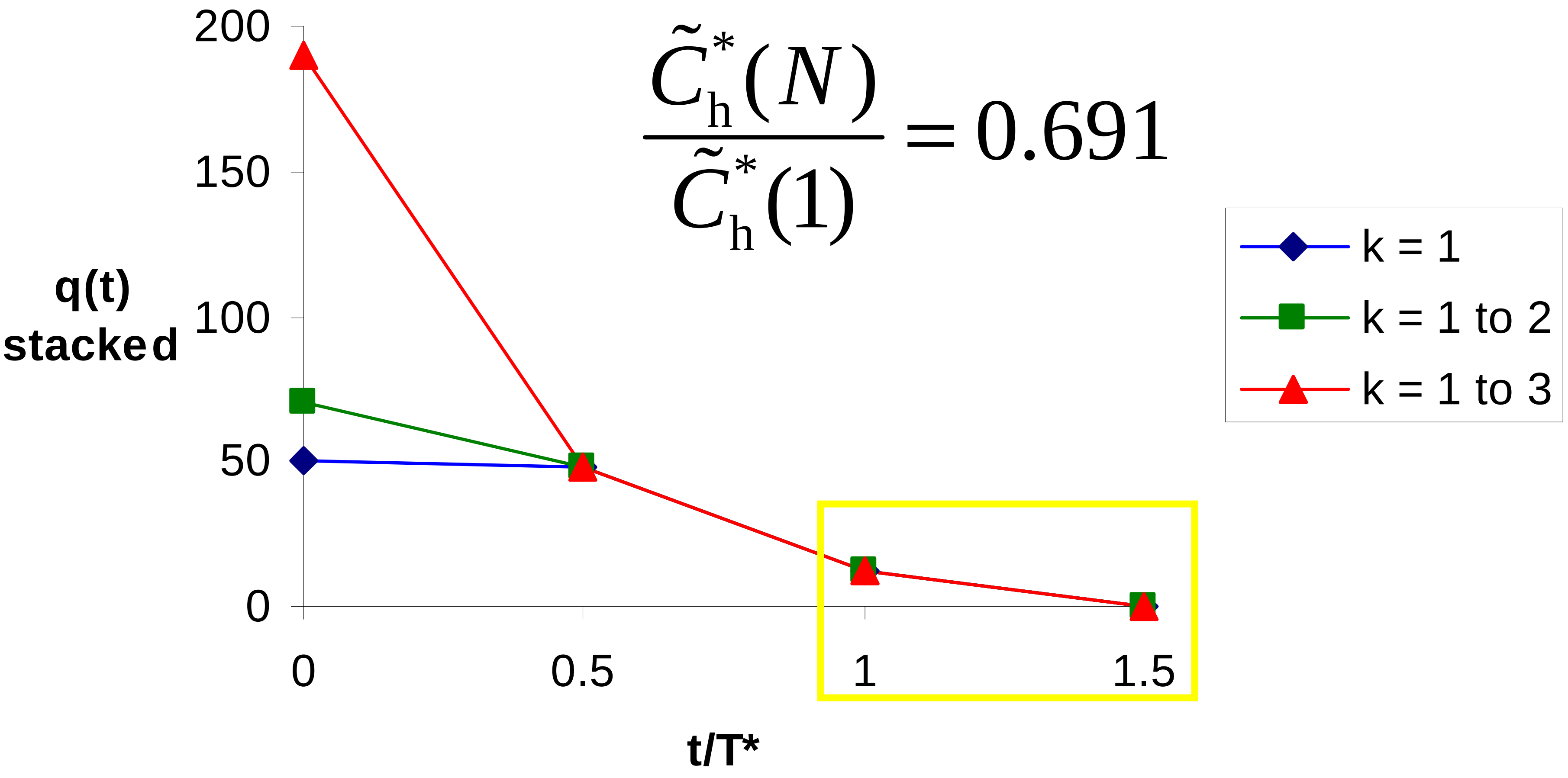
Feasible LP Solution for *N* = 2

$$\frac{\tilde{C}^*_{\mathrm{h}}(N)}{\tilde{C}^*_{\mathrm{h}}(1)} = 1.000$$

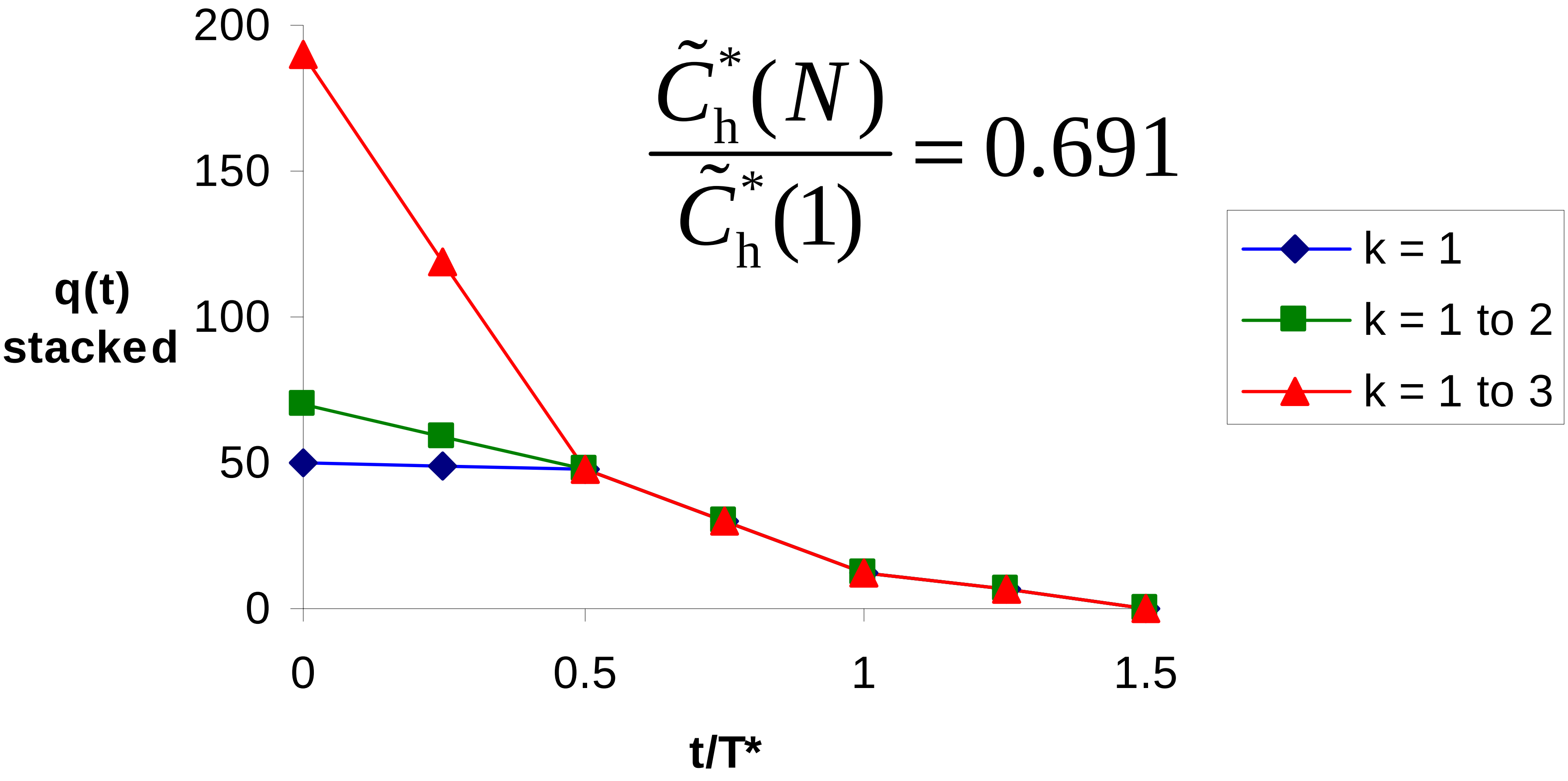


Optimal LP Solution for *N* = 3

$$\frac{\tilde{C}_{\mathrm{h}}^{*}(N)}{\tilde{C}_{\mathrm{h}}^{*}(1)} = 0.691$$

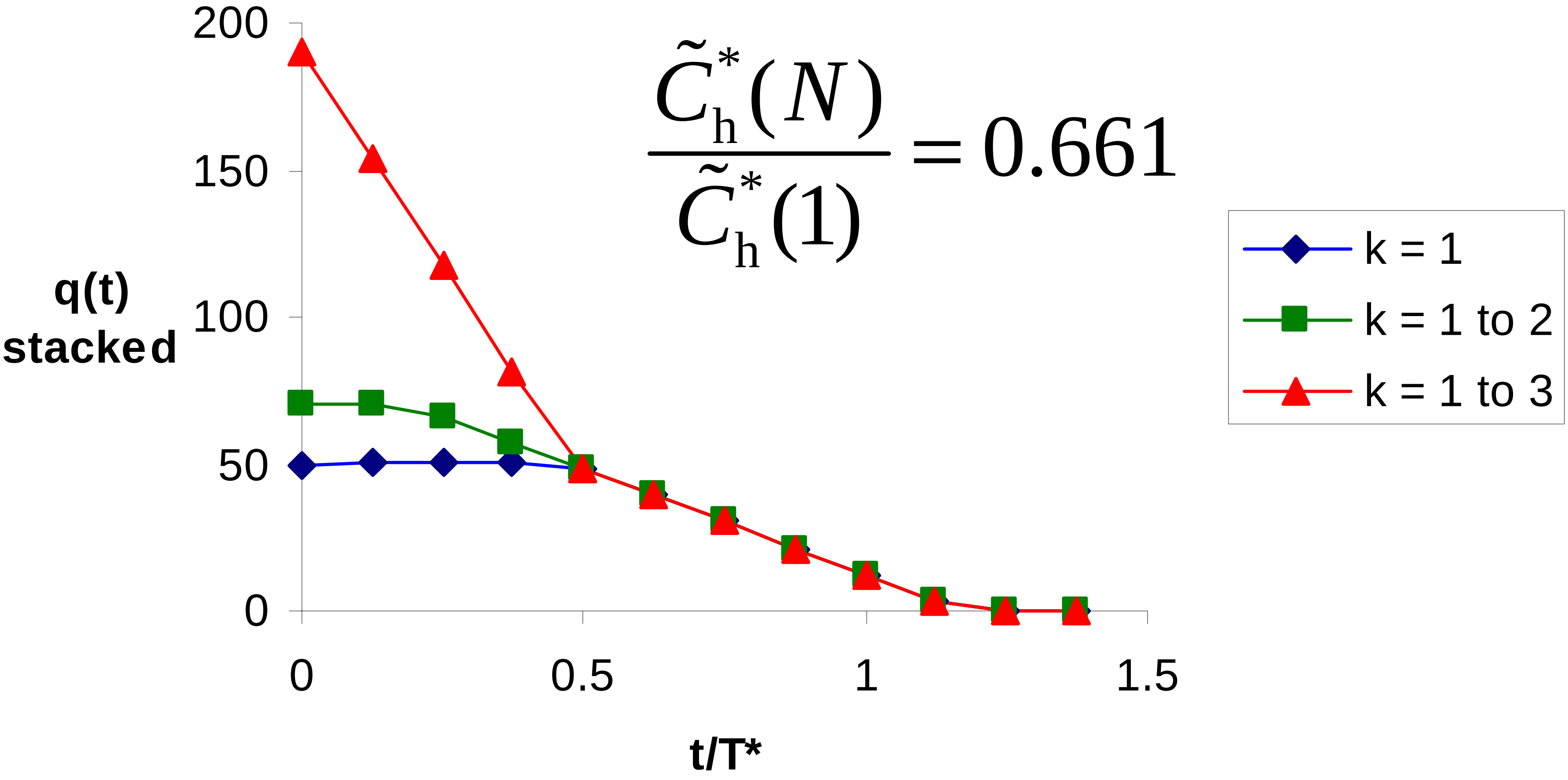


Feasible LP Solution for *N* = 6



$$\frac{\tilde{C}_{\mathrm{h}}^{*}(N)}{\tilde{C}_{\mathrm{h}}^{*}(1)} = 0.691$$

Optimal LP Solution for *N* = 10



$$\frac{\tilde{C}_{\mathrm{h}}^{*}(N)}{\tilde{C}_{\mathrm{h}}^{*}(1)} = 0.661$$

LP with All Variables Fixed Except at 1 Time Break Point

$$\min_{\underline{\tilde{\mathbf{q}}}(t_{n'}),\Delta t_{n'}} \tilde{C}_{\mathrm{h}}\left(\tilde{\mathbf{q}}(t_{n'}),\Delta t_{n'},\Delta\tilde{\mathbf{U}},\Delta\tilde{\mathbf{Y}},\tilde{\mathbf{Q}} \mid N,\left\{\underline{\tilde{\mathbf{q}}}(t_n)\right\}_{n\neq n'},\left\{\Delta t_n\right\}_{n\neq n'},T'\right)$$

$$=\frac{\underline{\mathbf{c}}^{\mathrm{T}}}{2}\left[\underline{\mathbf{q}}(t_0)\Delta t_1+\sum_{n=1}^{N-1}\underline{\tilde{\mathbf{q}}}(t_n)\left(\Delta t_n+\Delta t_{n+1}\right)\right]$$

s.t.

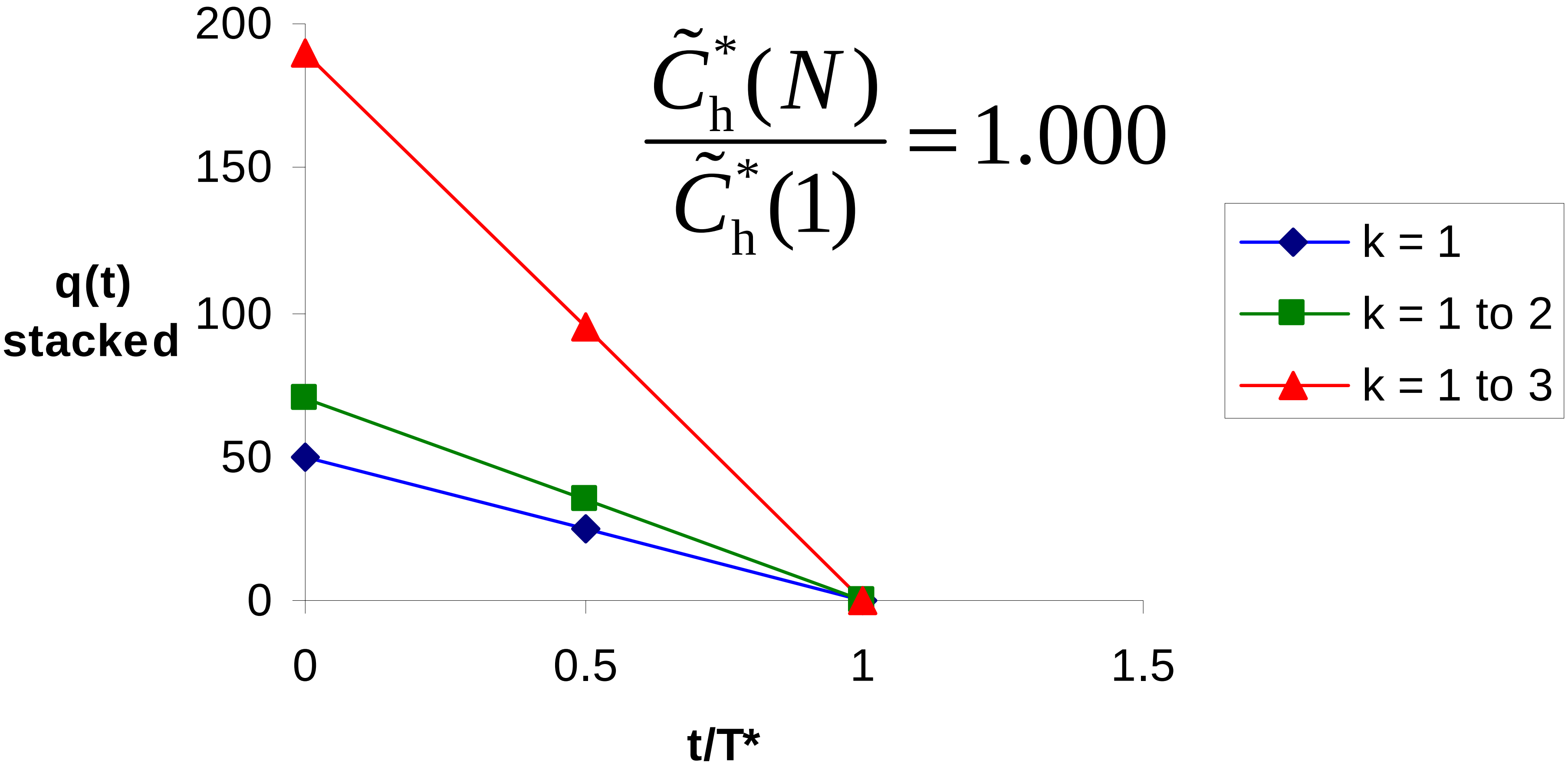
$$\underline{\tilde{\mathbf{q}}}(t_{n'+1})-\underline{\boldsymbol{\alpha}}T'\leq-\underline{\boldsymbol{\alpha}}\Delta t_{n'}+\underline{\tilde{\mathbf{q}}}(t_{n'})\leq\underline{\tilde{\mathbf{q}}}(t_{n'-1})$$

$$\mathbf{BD}(\mathbf{p})\underline{\tilde{\mathbf{q}}}(t_{n'-1})\leq\chi\Delta t_{n'}+\mathbf{BD}(\mathbf{p})\underline{\tilde{\mathbf{q}}}(t_{n'})\leq\mathbf{BD}(\mathbf{p})\underline{\tilde{\mathbf{q}}}(t_{n'+1})+\chi T'$$

$$-\left(\mathbf{I}_K-\mathbf{P}^{\mathrm{T}}\right)\underline{\tilde{\mathbf{q}}}(t_{n'})\leq\mathbf{0}_{K,1}$$

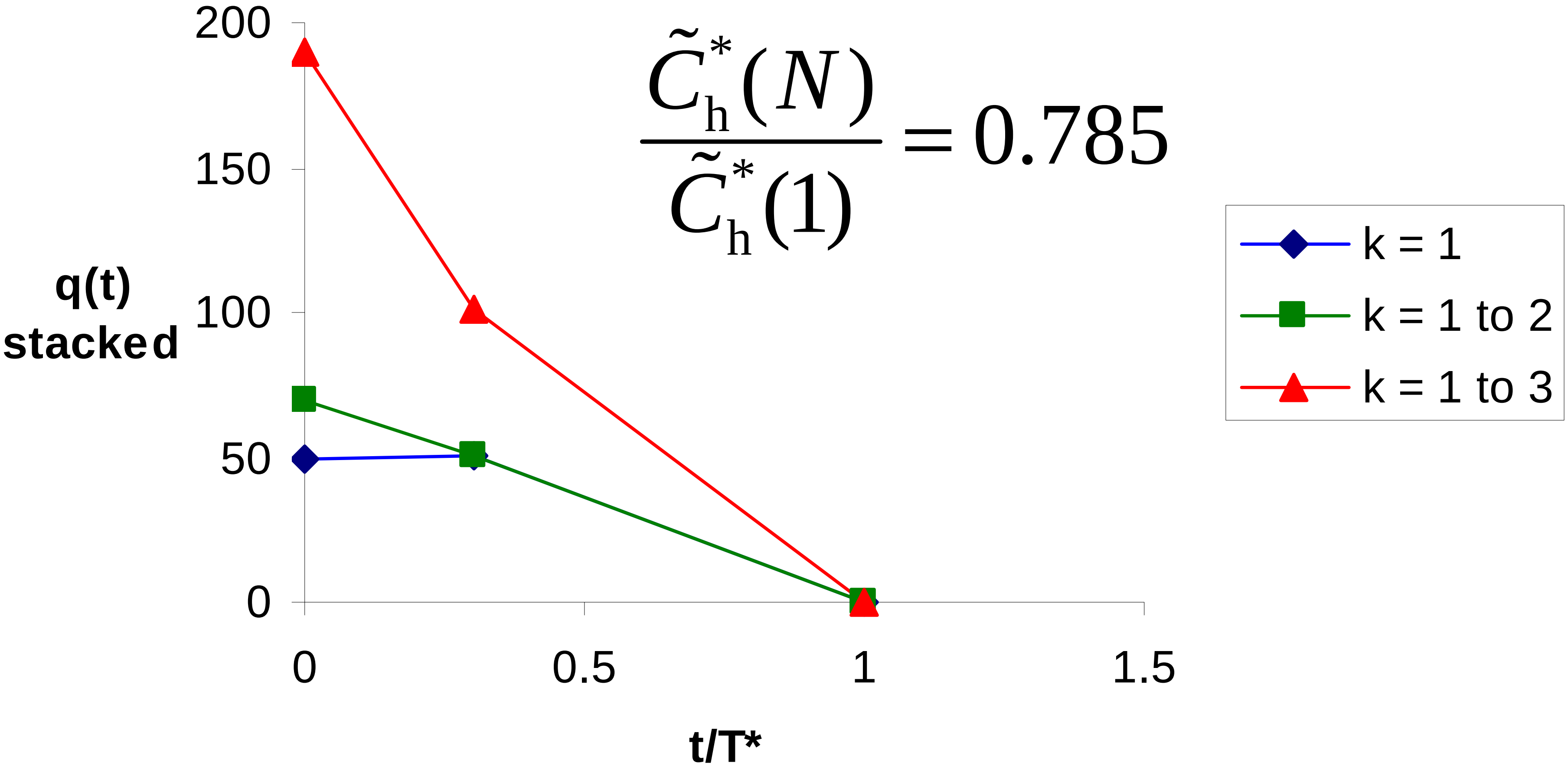
$$0\leq\Delta t_{n'}\leq T'$$

Feasible LP Solution for *N* = 2



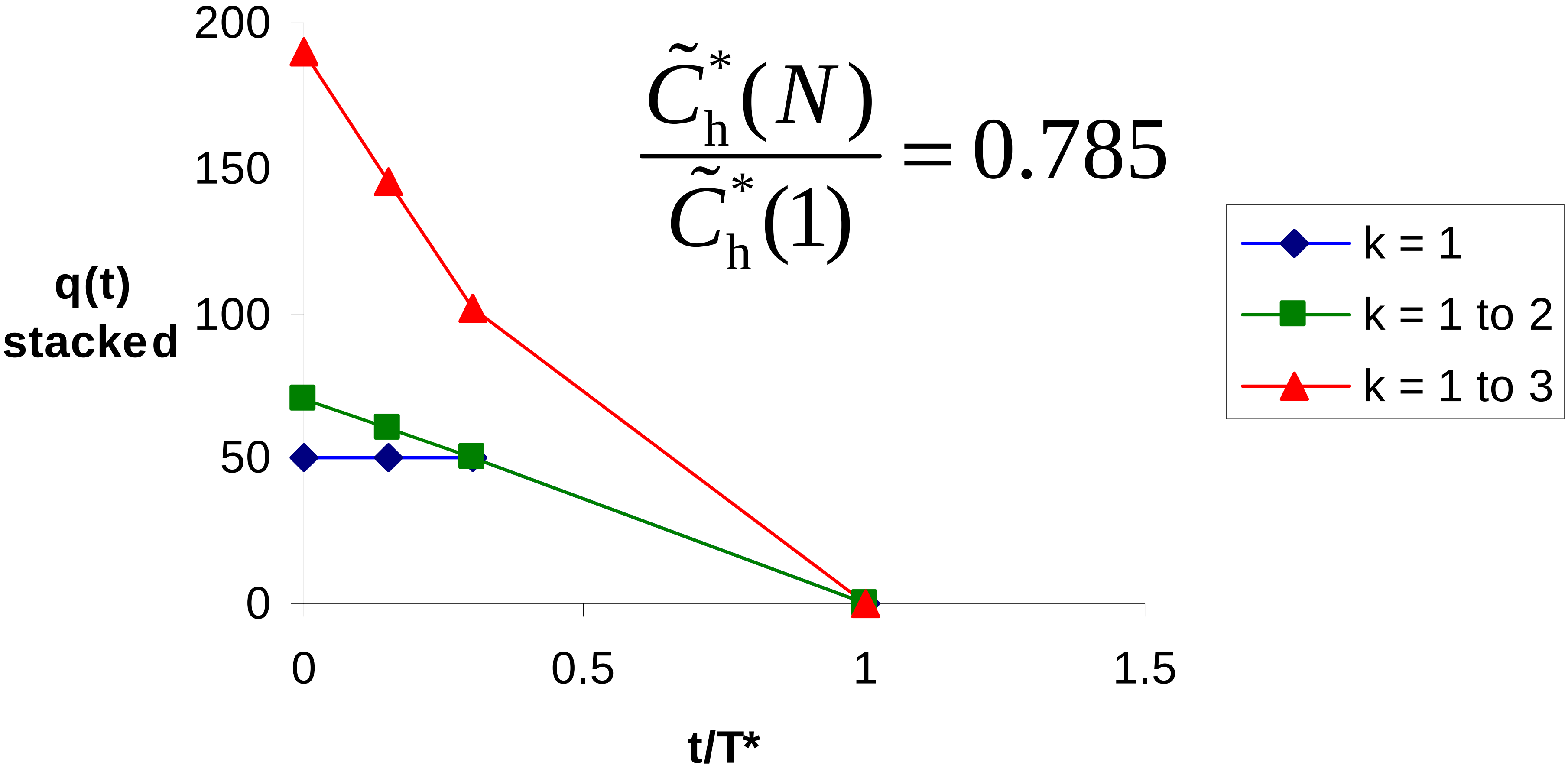
$$\frac{\tilde{C}_{\mathrm{h}}^{*}(N)}{\tilde{C}_{\mathrm{h}}^{*}(1)} = 1.000$$

Optimal LP Solution for *N* = 2



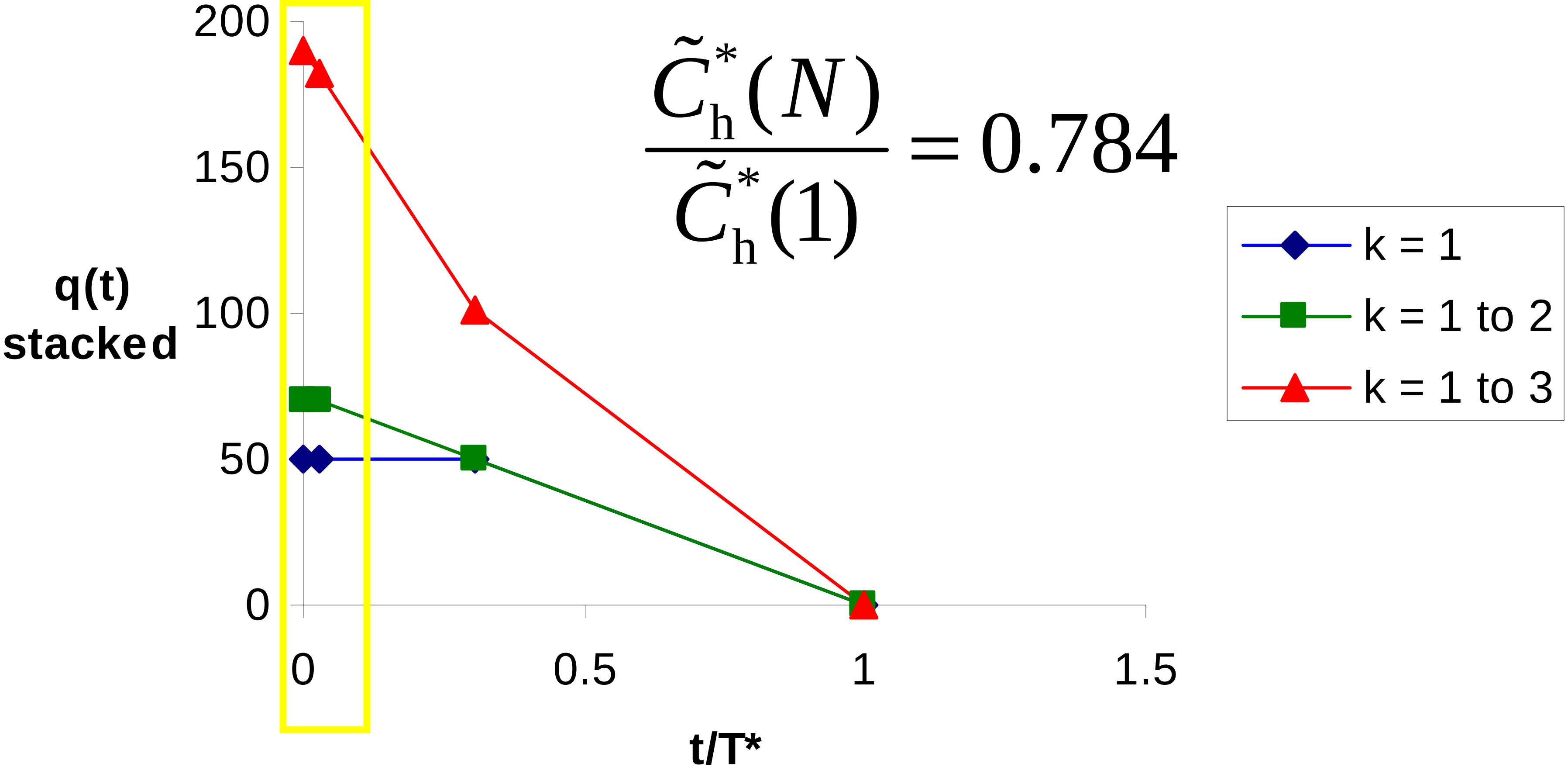
$$\frac{\tilde{C}_{\mathrm{h}}^{*}(N)}{\tilde{C}_{\mathrm{h}}^{*}(1)} = 0.785$$

Feasible LP Solution for *N* = 3



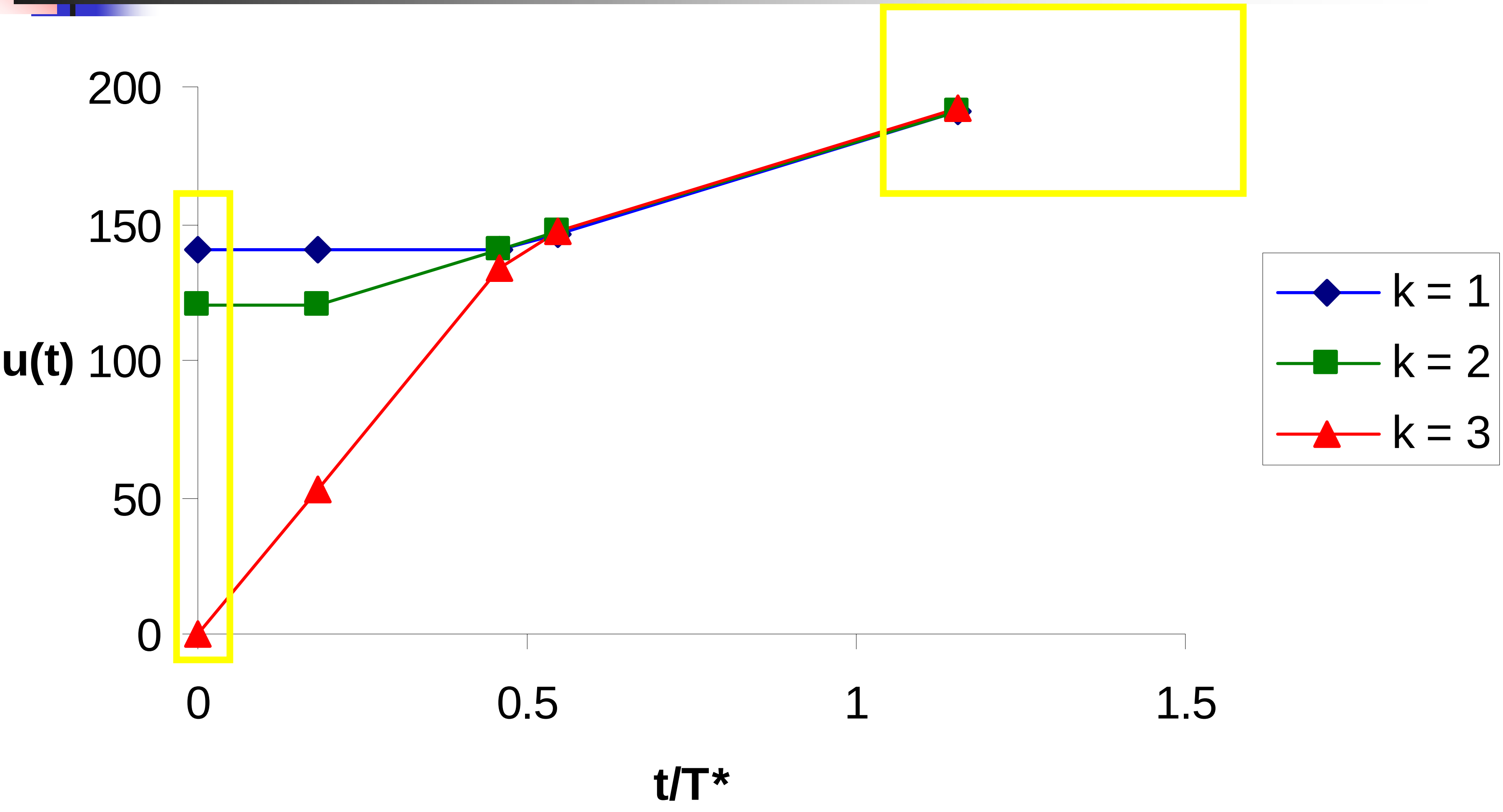
$$\frac{\tilde{C}_{\mathrm{h}}^{*}(N)}{\tilde{C}_{\mathrm{h}}^{*}(1)} = 0.785$$

Optimal LP Solution for *N* = 3



$$\frac{\tilde{C}^*_{\mathrm{h}}(N)}{\tilde{C}^*_{\mathrm{h}}(1)} = 0.784$$

Optimal QP (& SCLP) Solution: Cumulative Units Processed



Mixed Integer LP for Schedule Initialization

$$\min_{\hat{\mathbf{v}}(t_0^+)} \left\| \hat{\mathbf{v}}(t_0^+) - \tilde{\mathbf{v}}^*(t_1) \right\| \square$$

$$\text{s.t.} \quad \mathbf{B}\hat{\mathbf{v}}(t_0^+) \begin{cases} \leq \hat{\mathbf{m}} & \text{if idling is allowed} \\ \text{or} & \\ = \min\{\hat{\mathbf{m}}, \mathbf{Bq}(t_0)\} & \text{if idling is not allowed} \end{cases}$$

$$\hat{\mathbf{v}}(t_0^+) \begin{cases} \leq \mathbf{q}(t_0) \\ \text{and} \\ \in \square_+^K \end{cases}$$

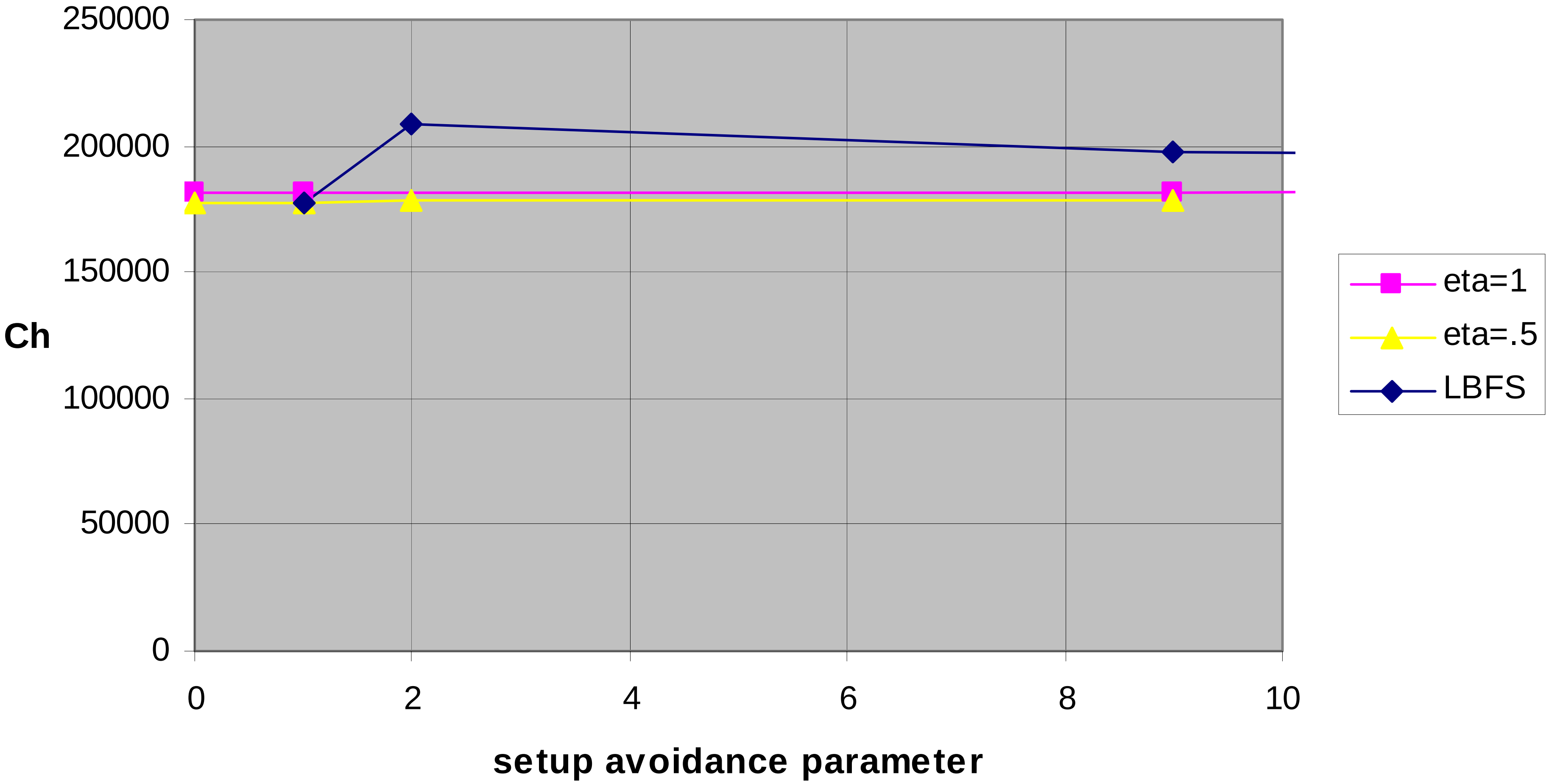
Issues with Adapting a Fluid Solution to a Real Factory

Fluid Analogy	*Factory Equivalent*
Bottlenecks	Limited machine capacity
Viscosity	Discrete (process batches, transport lots, ...)
Turbulence	Stochastic (random failures, service times, ...)
Bubbling	Sequence-dependent setups

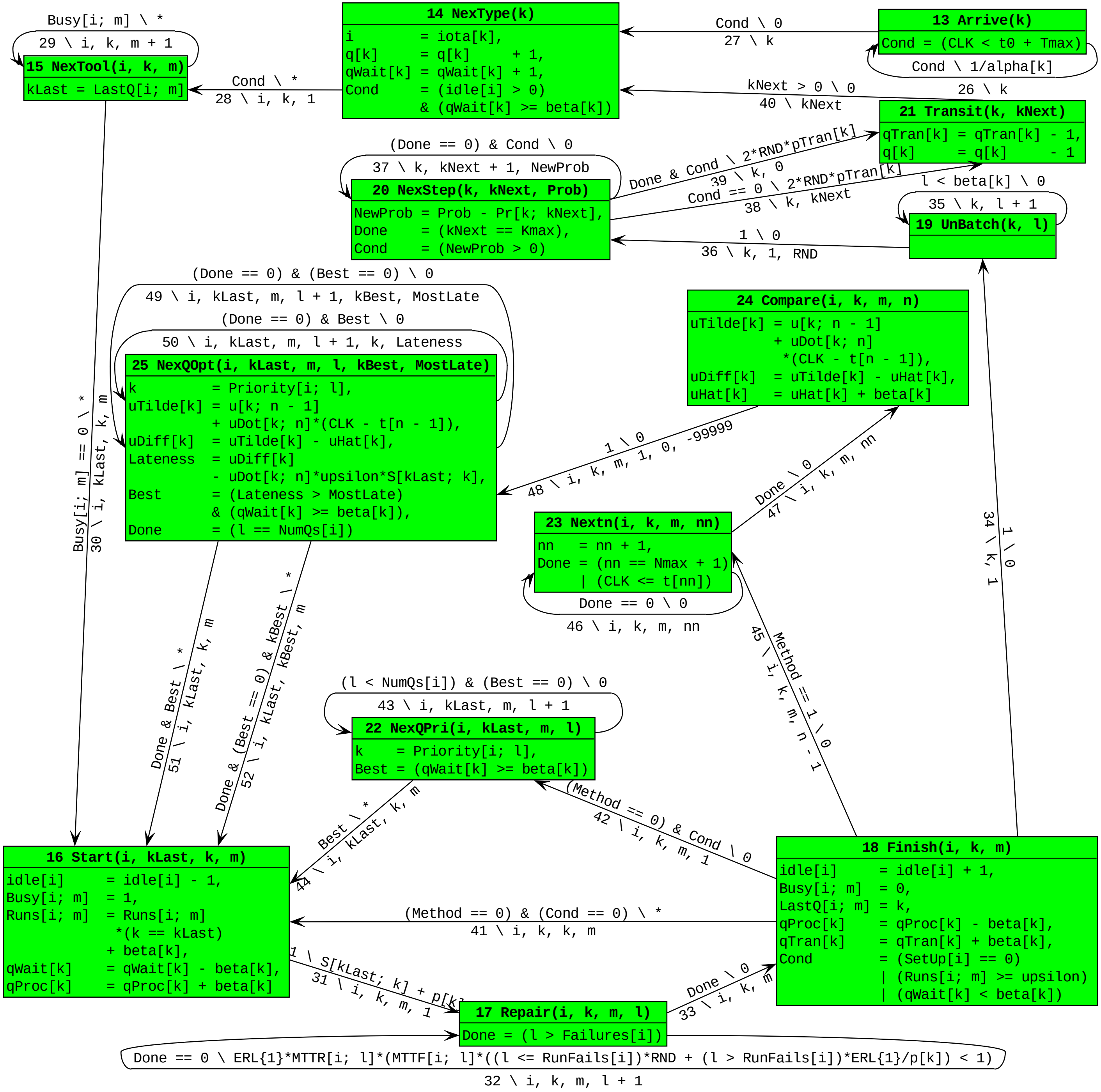
A Setup Avoidance Method

If the machine last processed jobs in queue *k* and the setup time required to next process jobs from any queue k' is $s_{k,k'}$ then the machine should process jobs from queue k' for which $\tilde{u}_k(t-\upsilon s_{k,k'})-\hat{u}_k(t)$ is largest.

Simulation Results

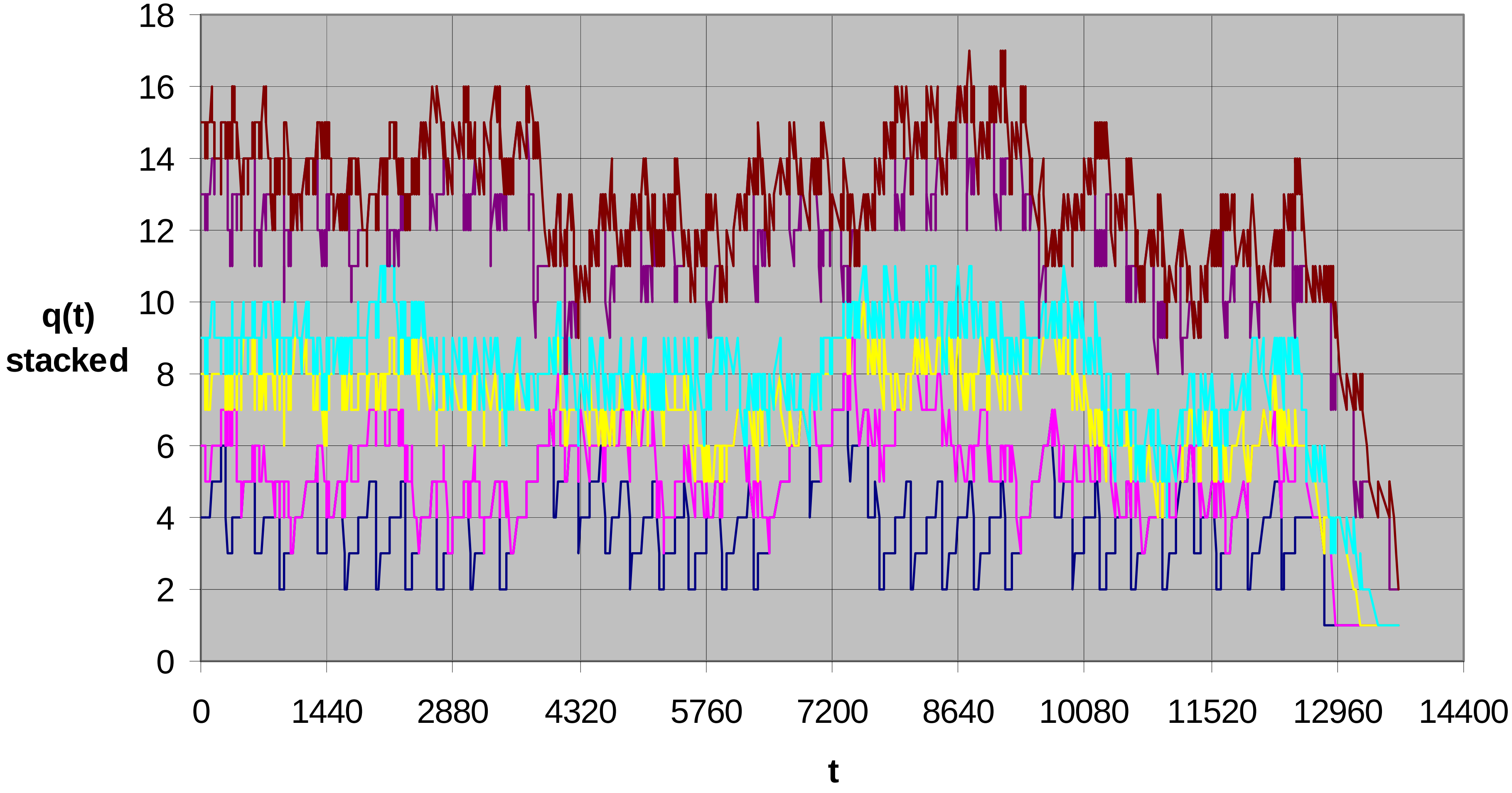


Simulation Event Graph



```
13 Arrive(k)
Cond = (CLK < t0 + Tmax)

14 NexType(k)
i         = iota[k],
q[k]      = q[k]      + 1,
qWait[k] = qWait[k] + 1,
Cond      = (idle[i] > 0)
          & (qWait[k] >= beta[k])

15 NexTool(i, k, m)
kLast = LastQ[i; m]

16 Start(i, kLast, k, m)
idle[i]      = idle[i] - 1,
Busy[i; m]   = 1,
Runs[i; m]   = Runs[i; m]
               *(k == kLast)
               + beta[k],
qWait[k]     = qWait[k] - beta[k],
qProc[k]     = qProc[k] + beta[k]

17 Repair(i, k, m, l)
Done = (l > Failures[i])

18 Finish(i, k, m)
idle[i]      = idle[i] + 1,
Busy[i; m]   = 0,
LastQ[i; m]  = k,
qProc[k]     = qProc[k] - beta[k],
qTran[k]     = qTran[k] + beta[k],
Cond         = (SetUp[i] == 0)
             | (Runs[i; m] >= upsilon)
             | (qWait[k] < beta[k])

19 UnBatch(k, l)

20 NexStep(k, kNext, Prob)
NewProb = Prob - Pr[k; kNext],
Done    = (kNext == Kmax),
Cond    = (NewProb > 0)

21 Transit(k, kNext)
qTran[k] = qTran[k] - 1,
q[k]     = q[k]     - 1

22 NexQPri(i, kLast, m, l)
k    = Priority[i; l],
Best = (qWait[k] >= beta[k])

23 Nextn(i, k, m, nn)
nn   = nn + 1,
Done = (nn == Nmax + 1)
     | (CLK <= t[nn])

24 Compare(i, k, m, n)
uTilde[k] = u[k; n - 1]
          + uDot[k; n]
           *(CLK - t[n - 1]),
uDiff[k]  = uTilde[k] - uHat[k],
uHat[k]   = uHat[k] + beta[k]

25 NexQOpt(i, kLast, m, l, kBest, MostLate)
k          = Priority[i; l],
uTilde[k] = u[k; n - 1]
           + uDot[k; n]*(CLK - t[n - 1]),
uDiff[k]  = uTilde[k] - uHat[k],
Lateness  = uDiff[k]
           - uDot[k; n]*upsilon*S[kLast; k],
Best       = (Lateness > MostLate)
           & (qWait[k] >= beta[k]),
Done       = (l == NumQs[i])
```

Simulation of Fluid Solution Adaptation



How Often to Recalculate Fluid Solution

- Daily or shiftly?
- Each time a lot arrives at an idle machine or a machine becomes available?
- Somewhere in between?

Conclusions

- Fluid models are potentially useful for finding good schedules for complex manufacturing systems.
- Large scale fluid model problems can be solved quickly.
- Translation issues are tricky but not insurmountable.