An Online Delay-Optimal Iterative Multiclass Scheduler for Generic M2M Uplink

Akshay Kumar, Ahmed Abdelhadi and Charles Clancy
Hume Center, Virginia Tech
Email: {akshay2, aabdelhadi, tcc}@vt.edu

Abstract—The uplink Machine-to-Machine (M2M) traffic is fairly heterogeneous in its delay-requirements both across the sensor nodes within a network and also across different M2M applications. Therefore, in this letter, we classify the uplink M2M traffic into multiple classes (based on delay requirements) and propose a delay-optimal multiclass packet scheduler that ensures proportional fairness of service to all classes. We accommodate the diverse delay requirements in our framework using generic sigmoidal utility functions of service delay. We note that any work-conserving scheduling policy can be realized by appropriately time-sharing between all possible preemptive priority policies. Then the optimal scheduler is determined iteratively by solving for the optimal fraction of time-sharing between all priority scheduling policies, that results in maximum system utility. The proposed scheduler can be implemented online with reduced computational complexity due to its iterative nature. Using Monte-Carlo simulations, we verify the optimality of the proposed iterative scheduler and show that it outperforms other state-of-the-art packet schedulers.

Index Terms—M2M, Multiclass, Delay-Optimal Scheduler, Convex Optimization

I. INTRODUCTION

Machine-to-Machine (M2M) communications is becoming increasingly popular due to its abundance of industrial-use cases such as monitoring power usage using smart meters, automation of manufacturing processes etc [1]. However, the delay-requirements of uplink traffic from the sensors to a Central Controller (CC) is highly application-specific. For instance, the monthly reporting of power usage from smart meters can tolerate delays of few weeks whereas industrial automation processes have stringent delay requirements of few milliseconds.

Most of the existing M2M packet schedulers are designed for specific wireless technology such as LTE and are heuristic schedulers without any guarantees of convergence (see [2] and references therein). Another line of work focuses on optimal scheduling algorithms specifically for real-time embedded systems (see [3] and references therein). Another drawback is that these schemes cannot be implemented online, making it difficult to adopt them for practical M2M systems.

In this letter, we propose an online delay-optimal multiclass packet scheduler for M2M uplink. We incorporate the delay-heterogeneity in traffic by classifying it into multiple classes; each parametrized by a set of packet delay parameters using a sigmoidal function. The objective of the scheduler is to maximize a proportionally-fair system utility function. We note that any work-conserving scheduling policy can be realized by appropriately time-sharing between all possible preemptive priority scheduling policies. Therefore, we determine the optimal scheduler by solving for the optimal fraction of time-sharing between priority scheduling policies that maximizes the system utility. We significantly reduce the computational complexity of the optimal scheduler by using an iterative algorithm instead of solving a single optimization problem.

Using Monte-Carlo simulations, we verify the optimality of the iterative scheduler and show that it outperforms various state-of-the-art schedulers such as First-Come-First-Serve (FCFS), Earliest-Due-Date (EDD) and priority scheduling policies. The proposed scheduler is agnostic to both the wireless technology used for the M2M uplink and the hardware-software system architecture. Since we obtain the analytical expressions for the optimal scheduler, it can easily adapt to time-varying system parameters such as packet arrival rate etc.

II. SYSTEM MODEL

Fig. 1 shows the system model with N sensors communicating to a CC on the M2M uplink. The delay-heterogeneous traffic (indicated by different colors of packets in Fig. 1) from the sensors enters one of the R (R ≪ N) queues at CC corresponding to the R delay classes. Assuming N is large enough and the packet arrivals from different sensors are independent of each other, the packet arrival process at the CC for class i can be be modeled as a Poisson process with rate $\lambda_i$. We model the service time at CC for packets of class i to be exponentially distributed with rate $\mu_i$. The total time spent by a packet in the system consists of transmission time at sensor, propagation and congestion delay from sensor to CC, queuing delay and service time at CC. In this work, we focus primarily on the queuing delay and service time at CC, assuming other components are relatively small.

A work-conserving scheduler does not result in server being idle while there are jobs in the queue waiting for service. Hereafter we drop the qualifier ‘preemptive’ without any ambiguity.

The packet size is usually small (few 100 bits) to ignore transmission time. At a relatively large, say 1 km separation between sensors and CC the propagation delay is 5 $\mu$s, hence it can be safely ignored. Lastly, we assume that the available wireless spectrum is large enough to permit a dedicated transmission channel for each sensor and thus we ignore congestion delay.

This research is based upon work supported by the National Science Foundation under Grant No. 1134843.
requirements by appropriately choosing the parameters
applications as shown in Fig. 3. A sigmoidal utility function is a good fit for delay-tolerant
sum of queuing delay and service time.

Now the queuing delay for each class depends upon the scheduling policy adopted at the CC. The scheduling policy at CC should be chosen so as to maximally satisfy the delay requirements of all traffic classes.

III. PROBLEM FORMULATION

We first use a generic sigmoidal function [4] to map the latency\(^4\) requirements of \(i\)-th class, \(l_i\), onto a utility function as,

\[
U_i(l_i) = 1 - c_i \left( \frac{1}{1 + e^{-a_i(l_i - b_i)}} - d_i \right)
\]

(1)

where, \(c_i = \frac{1 + e^{a_i b_i}}{e^{a_i b_i}}\) and \(d_i = \frac{1}{1 + e^{a_i b_i}}\). Note that \(U_i(0) = 1\) and \(U_i(\infty) = 0\). The parameter \(a_i\) is the utility roll-off factor and the inflection point for \(U_i\) occurs at \(l_i = b_i\).

The sigmoidal function is versatile to represent diverse delay requirements by appropriately choosing the parameters \(a\) and \(b\). For high \(a\) and low \(b\), the utility function becomes ‘brick-walled’ (see Fig. 2) and is a good fit for delay-sensitive applications. On the other hand, at low \(a\) and high \(b\), the sigmoidal utility function is a good fit for delay-tolerant applications as shown in Fig. 3.

\(^4\)We use the terms ‘delay’ and ‘latency’ interchangeably. Both refer to the sum of queuing delay and service time.

A. System utility function

For a given scheduling policy \(P\), we define a proportionally fair system utility function as,

\[
V(P) = \prod_{i=1}^{R} U_i(P),
\]

where \(U_i(P)\) is the average utility of traffic of \(i\)-th class in the steady state given as,

\[
U_i(P) = U_i \left( \lim_{T_i \to \infty} \frac{\sum_{j=1}^{M_i(T_i)} l_i^j(P)}{M_i(T_i)} \right),
\]

(3)

where \(M_i(T_i)\) is the number of packets of class \(i\) served in time \(T_i\) and \(l_i^j\) is the latency of the \(j\)-th packet of \(i\)-th class. The parameters \(\beta_i\) indicates the relative importance of utility of \(i\)-th class towards the system utility.

B. Iterative Optimization Problem

We now formulate an iterative optimization problem to solve for the optimal scheduling policy. Let \(A_r\) be set of any \(r\) (\(2 \leq r \leq R\)) classes and let \(B_r\) contain the remaining \(R-r\) classes, assigned higher priority than classes in \(A_r\). Now assume that in a sufficiently large time interval\(^5\), the CC serves \(i\)-th class with highest priority in \(A_r\) for \(\alpha_i\) fraction of the time. Then we can write the following set of equations for the average latency \(l_i\) of each class in \(A_r\),

\[
\begin{align*}
l_1 &= \alpha_1 l_{1,1} + \alpha_2 l_{1,2} + \cdots + \alpha_r l_{1,r}, \\
l_2 &= \alpha_1 l_{2,1} + \alpha_2 l_{2,2} + \cdots + \alpha_r l_{2,r}, \\
\vdots & \quad \vdots \quad \vdots \\
l_r &= \alpha_1 l_{r,1} + \alpha_2 l_{r,2} + \cdots + \alpha_r l_{r,r}.
\end{align*}
\]

(4)

Here \(l_{i,i}\) denotes the latency of the \(i\)-th class when it is served with highest priority. \(l_{i,j}\) denotes the optimal\(^6\) latency for the \(i\)-th class when serving the \(j\)-th class with highest priority in \(A_r\). Denote \(l_r = \{l_1, l_2, \ldots, l_r\}\) and \(l_{r,i}^{*} = \{l_{1,i}^{*}, l_{2,i}^{*}, \ldots, l_{r,i}^{*}\}\) \(\forall i\).

\(^5\)We assume the time interval under consideration is large enough to observe steady state queuing behavior.

\(^6\)Here optimal latency is obtained by optimizing over classes in \(A_r\) such that \(\alpha_i\) choose the optimal latency for each class.

Fig. 1. System Model with \(R = 3\) M2M traffic classes. The different packet colors of packets indicate different M2M delay classes.

Fig. 2. Utility function for delay-sensitive traffic: \(a = 50\) ms\(^{-1}\), \(b = 5\) ms.

Fig. 3. Utility function for delay-tolerant traffic: \(a = 0.3\) ms\(^{-1}\), \(b = 20\) ms.
Theorem 1. The optimization problem in (9) is convex.

Proof: The rigorous proof is removed due to lack of space. The proof outline follows from concavity of the objective function and linearity of the constraints in (9) and equivalence of (2) and (9) using the set of equations in (4).

IV. OPTIMAL SCHEDULER

We now solve the dual problem of (9) to determine the optimal solution. We first define the Lagrangian as,

\[ L(\tilde{\alpha}_r, \eta) = \sum_{i=1}^{r} \beta_i \log(U_i(\tilde{\alpha}_r)) - \eta \left( \sum_{i=1}^{r} \alpha_i - 1 \right), \]

where \( \eta \geq 0 \) is a Lagrange multiplier. The dual problem is formulated as follows,

\[ \min_{\eta} \max_{\tilde{\alpha}_r} L(\tilde{\alpha}_r, \eta) \]

subject to \( \eta \geq 0, \alpha_i \geq 0, \forall i \in \{1, 2, \cdots, r\} \).

At the optimal solution \( (\tilde{\alpha}_r^*, \eta^*) \), we have,

\[ \frac{\partial}{\partial \alpha_i} \left[ \log(V(\tilde{\alpha}_r)) - \eta \left( \sum_{j=1}^{r} \alpha_j - 1 \right) \right]_{\tilde{\alpha}_r = \tilde{\alpha}_r^*} = 0. \quad (12) \]

\[ \Rightarrow \sum_{j=1}^{r} \beta_j \frac{U_j(\tilde{\alpha}_r)}{U_j(\tilde{\alpha}_r)} U_j(\tilde{\alpha}_r) \bigg|_{\tilde{\alpha}_r = \tilde{\alpha}_r^*} = \eta^* \forall i. \quad (13) \]

Using (1) and (4), we have,

\[ \frac{\partial}{\partial \alpha_i} U_j(\tilde{\alpha}_r^*) = \frac{-c_j \beta_j a_j e^{-a_j(l_j-b_j)} l_j^*_i}{(1 + e^{-a_j(l_j-b_j)})^2} \forall i, \]

where \( l_j^*_i = l_j^* \) when \( j = i \).

Substituting (14) in (13), followed by some algebraic manipulations, we get,

\[ \sum_{j=1}^{r} \frac{k_i}{1 + e^{-a_j(l_j-b_j)}} |_{\tilde{\alpha}_r = \tilde{\alpha}_r^*} + \eta^* = 0 \forall i. \quad (15) \]

where \( k_i = \beta_j a_j l_j^* \) is a constant.

Also, at the optimal solution \( (\tilde{\alpha}_r^*, \eta^*) \), we have,

\[ \frac{\partial}{\partial \eta} \left[ \log(V(\tilde{\alpha}_r)) - \eta \left( \sum_{j=1}^{r} \alpha_j - 1 \right) \right]_{\tilde{\alpha}_r^*, \eta^*} = 0, \quad (16) \]

\[ \Rightarrow \sum_{j=1}^{r} \alpha^*_j = 1. \quad (17) \]

We determine the optimal solution by solving for \( (\tilde{\alpha}_r^*, \eta^*) \) using (15) and (17) simultaneously. The constraint \( \alpha^*_i \geq 0 \) is enforced by setting \( \alpha^*_i = 0 \) if it is negative and then resolve for \( \tilde{\alpha}_r^* \). Now the optimal latency for \( r \) class subsystem, \( \tilde{l}^*_r \), is obtained by setting \( \tilde{\alpha}_r = \tilde{\alpha}_r^* \) in (4).

The iterative algorithm for determining the optimal multiclass scheduler is described in Algorithm 1. The function \texttt{OPT}\texttt{SCH} is initialized with \( r = R \) and \( S_R = \emptyset \). At the \( r \)-class recursion, we iterate through each class in set \( A_r \), assign it as highest priority in \( A_r \), and then solve the resultant \( r - 1 \) class subsystem using \( A_{r-1} \). This recursion is performed until \( r = 2 \) for which the optimal solution is determined using procedure outlined in Section IV. The output of \texttt{OPT}\texttt{SCH} for \( r - 1 \) class subsystem \( (\tilde{l}^*_{r-1}, \tilde{\alpha}^*_{r-1}) \), is used to determine the optimal scheduler for \( r \) class subsystem.

Algorithm 1 Proposed Iterative \( r \)-Class Scheduler

```
function \texttt{OPT}\texttt{SCH}(\tilde{\alpha}_r)
    if length(\( A_r \)) > 2 then \( \triangleright \) \( A_r = \{1, 2, \cdots, R\} \setminus S_r \)
        for \( i \in A_r \) do
            if \( \tilde{l}^*_{r-1} \) \( \triangleright \) \( \texttt{OPT}\texttt{SCH}(\{S_r, i\}) \)
                Set \( \tilde{l}^*_{\{i\}} = \tilde{l}^*_{r-1} \) and determine \( \tilde{l}^*_{i, i} \) using Eq (5).
            else \( \triangleright \) For \( r = 2 \)
                Use Eq (5) and (7) for \( l_{1,1}^*, l_{1,2}^*, l_{1,2}^*, l_{2,1}^* \).
        Use procedure in Section IV to solve for \( \tilde{l}^*_{r} \) and \( \tilde{\alpha}^*_{r} \).
    return \( (\tilde{l}^*_{r}, \tilde{\alpha}^*_{r}) \).
```

A. Complexity Analysis

\textbf{Direct optimization:} For a \( R \) class system, there are \( \Omega R! \) unique priority orders among the classes and we need to determine the optimal fraction of time \( \alpha^*_i \) the scheduler serves using the \( i \)-th priority order. Thus, we need to solve a set of \( R! + 1 \) nonlinear equations to determine \( \tilde{\alpha}^*_{i} \). Hence, the computational complexity of determining the optimal scheduler is enormous, even at moderately large values of \( R \).
Iterative optimization: In this case, we solve multiple optimization problems for \( r \) class subsystems (\( \forall 2 \leq r \leq R \)), with \( r \) variables. Thus we solve a set of \( r + 1 \) nonlinear equations. Since the priority order among the classes in \( A_r \) does not effect the latency of classes in \( A_r \), the number\(^3\) of \( r \) class optimization problems are \( R!/r!(R-r)! \). Thus, the iterative algorithm scales well with the number of classes by solving multiple but quite small optimization problems. This complexity reduction using iterative optimization is illustrated in Fig. 4 that plots the maximum number of simultaneous nonlinear equations that need to be solved in both schemes as the number of classes \( R \) are increased.

Clearly, the maximum complexity of an optimization problem for iterative optimization is negligible compared to the direct optimization, even though it requires us to solve a large number of optimization problems.

V. RESULTS

We now use Monte-Carlo simulations to compare the delay-performance of the proposed scheduler with various state-of-the-art schedulers. Fig. 5 shows the plot of system utility for a 4 class system as the arrival rate of class 1 is varied. The system utility for all the schedulers decreases with increase in \( \lambda_1 \) due to the increased traffic at CC. We verify the optimality of the iterative scheduler as it matches with the simulation and analytical result for the Direct optimization problem. Thus the iterative scheduler does not loose in optimality; however it has significantly low computational complexity compared to Direct optimization problem.

Now since the proposed scheduler is optimal, it outperforms the FCFS, EDD and priority scheduling policies. We note that at low \( \lambda_1 \), priority to class 4 is a near-optimal policy as it has most stringent latency requirements \( (a_4 = 0.7, b_4 = 1, \beta_4 = 0.8) \). The loss in optimality due to the resultant FCFS service among non-class 4 traffic is minimal because the delay-requirements of class 2 and 3 are similar and \( \lambda_3 \sim 0 \). But at high \( \lambda_1 \), priority to class 4 is far from optimal because the FCFS service among non-class 4 traffic degrades the system utility significantly. This is because class 1 is most delay-tolerant and has least \( \beta_1 \), so it should be given least priority.

VI. CONCLUSIONS

In this letter, we proposed an online delay-optimal multiple class packet scheduler for a heterogeneous M2M uplink. Based on the delay-requirements, we classify the uplink traffic into multiple classes each characterized by a unique set of sigmoidal function parameters. We utilize the fact that any work-conserving scheduling policy can be realized by appropriately time-sharing between all preemptive priority scheduling policies. Then the optimal scheduler is determined by solving for the optimal fraction of time-sharing between priority scheduling policies so as to maximize a (proportionally-fair) system utility function. We significantly reduce the computational complexity of optimal scheduler by iteratively solving multiple but small optimization problems rather than a single big optimization problem. Using Monte-Carlo simulations, we verified the optimality of the iterative scheduler and showed that it outperforms the state-of-the-art schedulers such as FCFS, EDD and priority scheduling policies.

REFERENCES