Adaptive multi-model monitoring of recurrent mobility patterns

L. Papachristoforou*, P. Kolios§, C. Panayiotou§, and G. Ellinas§

***Abstract*—Multi-model event-triggering is a highly promising technique for efficient monitoring of processes where instead of continuous or even periodic triggering of events, communication and control is only applied after some event interrupt. In this work we investigate an adaptive multi-model monitoring technique whereby a local host that switches between the observed models informs remote hosts of these events which in turn adapt their predictions to reduce prediction error and minimize unnecessary triggering events and future model switching, thereby reducing energy consumption and communication bandwidth.**

The adaptive technique is examined under a real public transport bus service scenario, where local and remote hosts use a set of mobility models to track travel times and update their arrival schedules according to detected deviations, i.e., event interrupts.

***Index Terms*—Event triggering, Energy Efficiency, Multi-Model Event Triggering, IoT devices**

I. Introduction

The use of interconnected devices is rapidly increasing mainly due to their ability to execute useful functions without human intervention. However, most of these devices are battery operated and thus have limited operating times. As one of the most power-hungry features in these devices is communication, the literature examines and implements algorithms to limit communication from/to these devices without affecting their performance and functionality. Event-triggering (ET) algorithms are able to achieve this objective by limiting communication to specific event interrupts. In the meantime, when no communication is required, the communication circuitry can go to sleep and the device limits its functionality to local operations.

Under this setting, ET algorithms aim to first establish models to characterize the system. A local host monitors the system while remote hosts maintain an estimate of the system state based on the active model. Both the local and remote hosts possess one or multiple models of the system. The remote host uses the active model to estimate the system state when no communication between local and remote hosts exists. The local host compares the true state to the active operating model and decides whether to generate an event interrupt for resynchronization with the remote host and for switching to a more accurate model. By representing the system state with multiple models, the system dynamics are more accurately captured, and at the same time the number of resynchronization events is decreased.

In this work, we demonstrate how such a multi-model architecture can be realized through a probabilistic data-driven framework and investigate how switching between models can be done in an adaptive way to minimize event triggering. Specifically our contributions can be summarized as follows:

- A probabilistic framework is developed to estimate mobility patterns and a multi-model technique is devised to improve estimation accuracy.
- Three monitoring schemes are elaborated that use the proposed multi-model architecture to monitor mobility changes. The first scheme assumes no memory in the sequence of models used by preceding vehicles, the second scheme assumes that the most-probable model is always used across segments of the route and the last scheme assumes that subsequent vehicles use the latest sequence of models observed by the preceding vehicle in the particular route.
- An extensive performance evaluation of the three schemes is conducted using real mobility traces collected from buses operated by the public transport authority in Nicosia, Cyprus.

The rest of the paper is organized as follows. Related work is included in Section II and our novelties and contributions to the state-of-the-art are emphasized. The multi-model event triggering framework is detailed in Section III, while the two alternative schemes are elaborated in Sections IV and V respectively. Experimental results are presented in Section VI and Section VII concludes this work.

II. Related Work

The Internet of Things (IoT) is an emerging network that is dominated by a large number of interconnected physical devices, capable to operate without human intervention. Their ability to automatically perform functions that augment and aid the quality of life, make the IoT an attractive proposition. Consequently, the number of IoT devices is predicted to increase dramatically in the near future; according to CompTIA's research [1] 50 billion IoT devices would be connected to the Internet by 2020, while Cisco estimates that this number will increase to a staggering 11 trillion devices by 2025 [2].

Nevertheless, IoT devices currently face some physical constraints that limit their use, including their limited energy resources. This is especially true for mobile and embedded devices where access to power for recharging is not readily

*L. Papachristoforou is with the Department of Electronic and Electrical Engineering, Imperial College London lp1514@ic.ac.uk.
§P. Kolios, C. Panayiotou, and G. Ellinas are with the KIOS Research Center for Intelligent Systems and Networks and the Department of Electrical and Computer Engineering, University of Cyprus {pkolios, christosp, gellinas}@ucy.ac.cy.

978-1-5386-4980-0/19/$31.00 2019 European Union

available [3], [4]. A popular example is mobility trackers (usually GPS devices) which are installed to monitor vehicle movements in fleet management systems and in public transport systems to inform of the arrival of trucks/buses to specific stations. In the latter case, models of the recurrent mobility behavior of buses serving particular routes can be derived and used to predict future vehicle positions and allow for estimates in the arrival times at bus stops. Accurate movement predictions, in turn, will alleviate the need for continuous communication to track a particular vehicle. Consequently, as elaborated in [5], several different model-based solutions were examined in the literature in order to minimize communication and hence the energy consumption of these devices (enhancing their lifespan while at the same time preserving tracking accuracy).

Trace compression is a popular approach which is extensively examined in the literature [6]- [7]. This approach aims at reducing the size of the trajectory data resulting in improved energy efficiency during communication, without sacrificing the quality of information being transmitted.

Event triggering is an alternative technique that has been developed to limit communication between devices while preserving high tracking accuracy [5], [8], [9]. Several strategies to handle ET signals are proposed, as is the case in [10], while many centralized, distributed and cooperative ET architectures are examined in the literature [11]–[13].

In this paper, we elaborate on the multi-model event triggering (MMET) technique, and two extensions are then investigated, namely the baseline MMET (BMMET) and the proactive MMET (PMMET). The BMMET technique aims to utilize the various models generated to eliminate the communication due to synchronization events between the remote and local hosts. On the other hand, the PMMET technique aims to develop a dynamic update protocol of the behavior models in order to enable their proactive change. For validation, the performance of all three schemes is subsequently evaluated and compared against each other.

III. Multi-Model ET (MMET)

In this section, we elaborate on the proposed multi-model ET technique. To do so, we first develop a probabilistic framework to generate a set of behavior models, and then use these models to estimate the evolution of the system. At each time instance, both the local and remote hosts operate on a single model, ξ, but switch between different behavior models when deemed necessary in order to improve tracking accuracy and minimize further triggering.

The practical scenario we consider is that of monitoring a fleet of buses serving a particular route of the public transport network as detailed in [14]. A predetermined model or set of models of the bus mobility is generated and used to track the movement of each bus along that route. As long as movement of a bus matches its operating model, no communication between the two hosts is required. However, when a deviation between the model and the actual movement exceeds a threshold, an event interrupt is triggered. The local host then simply sends a resynchronization event to the remote monitoring host and both entities switch to a new model (if needed). The objective of this procedure is to track the movement of the buses accurately with minimal communication between the local and remote hosts.

A. Multi-model probabilistic framework

To build the relevant mobility models, a data-driven approach is pursued. Real data provided by the Transportation Organization of Nicosia (OSEL) in Cyprus is used as our benchmark. The data consists of locations (longitude, latitude) and timestamps for 3 buses operating on a particular service (in this case, Route 150 that consists of $\mathcal{B} = 42$ bus stops). The training and test sets consists of $S^{train} = 426$ and $S^{test} = 107$ bus trip traces, respectively.

Using these traces, probabilistic mobility models are derived along a particular route. Let $S_{i,j}$ be the travel time samples between bus stops $\{i, j\} \in \mathcal{B}$, where $\mathcal{B}$ is the set of predetermined bus stops. The travel time is the measured elapsed time between the timestamped samples received at bus stops i and j. Then, the first sample moment $\theta_{ij}(N)$ after N sample measurements is given as follows:

$$\theta_{ij}(N) = \frac{1}{N}\sum_{n=1}^{N} t_{ij}(n),\ \forall\, \{i, j\} \in \mathcal{B} \tag{1}$$

while higher order moments (i.e., $\theta_{ij}(N)^k = \frac{1}{N}\sum_{n=1}^{N}[t_{ij}(n)]^k$ for the k^{th} moment) improve the knowledge with regards to the actual distribution of the random variable. Having this distribution with respect to travel times allows for the computation of probabilistic estimates on the future travel times, while bounds can be set to detect unanticipated changes and trigger out-of-bound events. Hence an event is defined as follows:

Definition 1: An event is triggered whenever the actual travel times measures across a particular segment $i \mapsto j$ exceed the set bounds of the mobility model being used.

In practice multiple models can be defined $\mathcal{M} = \{1, \ldots, M\}$, to improve tracking accuracy. To define those model the travel time distribution $S_{i,j}$ along segment $i \mapsto j$, $\{i, j\} \in \mathcal{B}$ is split into M equal-sized parts as shown in fig. 1 for the case of $M = 4$. The blue dotted lines correspond to the mean time of each model. For instance, $\tau_{1,4}(2)$ corresponds to the mean of the second model of the distribution $S_{1,4}$.

B. Event Detection and Event Handling

As introduced above, event detection can be achieved by setting a bound α, around the mean $\tau_{i,j}(m), m \in \mathcal{M}$ of the distribution $S_{i,j}$. Any measurement recorded outside that bound would trigger an event interrupt. In the latter case, the operating model will switch to model $\xi \in \mathcal{M}$ for which its mean, $\tau_{i,j}(\xi)$ is the closest to the observed measurement. The local host will additionally inform remote hosts about the event and switching decision. Mathematically this switching can be computed simply by $\min_{\xi\in\mathcal{M}} |\tau_{i,j}(\xi) - t_{ij}|$. The intuition of switching to this new model ξ is that any subsequent near-future mobility patterns will be better captured using the

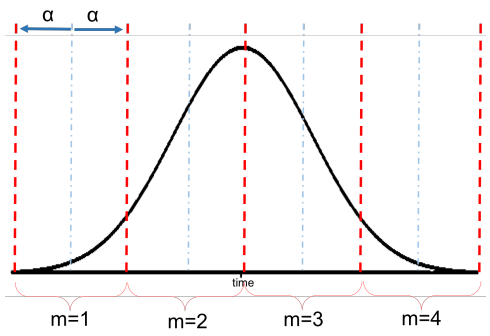

Fig. 1. Generation of $M = 4$ models for the distribution $S_{1,4}$. The blue dotted lines correspond to the mean of each model, $\tau_{1,4}(m = \mathcal{M})$.

mobility predictions of ξ and thus event triggering will be minimized.

As theoretically proven in [14], the probability of a particular number of events, $P_T(E, \alpha)$, depends on the total number of events, E, and the violation bound, α. For completeness, we summarize below how the total probability is computed.

To enumerate all combinations of events that can occur along an arbitrary route, the binomial coefficients are computed. Let matrix entry $\mathbb{C}_{ce}$ determine the $e^{\text{th}}, e = 1, \ldots, E$ deviation event that occurs in the c^{th} instantiation of events, $c = 1, \ldots, C$, where $C = \frac{B!}{(B-1)!E!}$ is the number of all possible combinations that can occur. Then, the probability of an event occurring at $j \in \mathcal{B}$, given that the previous event occurred at segment $i \in \mathcal{B}$, can be expressed as follows:

$$P(i, j) = \bar{P}(i, i+1) \times \ldots \times \bar{P}(i, j-1) \times (1 - \bar{P}(i, j)) \quad (2)$$

where the probability $\bar{P}(i, j)$ of no event triggered when traversing segment $i \mapsto j$ is $\bar{P}(i, j) = P(\tau_{ij} - \alpha \leq \mathcal{S}_{ij} \leq \tau_{ij} + \alpha)$.

Let $\mathcal{S}_{ij} = \{t_{ln}, l = (i \mapsto j), n = 1, \ldots, N\}$ include all samples observed between stops $i \mapsto j$, $\{i, j\} \in \mathcal{B}$. Then, border cases between event interrupts should also be considered. The leftmost border deals with the probabilities from the start of the route up to the 1^{st} event. Similar to (2), these probabilities can be expressed as follows:

$$P(1, i) = \bar{P}(1, 2) \times \ldots \times \bar{P}(1, i-1) \times (1 - \bar{P}(1, i)) \quad (3)$$

The rightmost border deals with the probability of no event occurring from the last event to the end of the route. This probability can be expressed as follows:

$$P(j, |\mathcal{B}|) = \bar{P}(j, j+1) \times \bar{P}(j, j+2) \times \ldots \times \bar{P}(j, |\mathcal{B}|) \quad (4)$$

Thus, the total probability would then be:

$$P_T(E, \alpha) = \sum_{c=1}^{C} \prod_{e=1}^{E-1} [P(\mathbb{C}_{ce}, \mathbb{C}_{ce+1})] \times P(1, \mathbb{C}_{c1}) \times P(\mathbb{C}_{cE}, |\mathcal{B}|) \quad (5)$$

where $P(\mathbb{C}_{ce}, \mathbb{C}_{ce+1})$ is given by equation (2) and the last two terms are given by equations (3) and (4), respectively.

Intuitively, different values of parameter α result in different numbers of out-of-bound events, E. For instance, a large value of α will lead to a small number of out-of-bound events and similarly, a small value of α will lead to a large number of out-of-bound events. Our objective is to tune α, such that the number of events generated is not larger than the desired number of events, K. Note that K is within the range of $[0, (B-1)]$. These observations enable us to calculate the Cumulative Distribution Function (CDF) of $P_T(E, \alpha)$ as follows:

$$P_{CDF}(K, \alpha) = \sum_{E=0}^{K} P_T(E, \alpha), \quad \alpha \in \mathbb{R}, \alpha \geq 0 \quad (6)$$

$P_{CDF}(K, \alpha)$ is defined as the probability of generating no more than K out-of-bound events, given a violation bound α. The following algorithm is utilized to calculate $P_{CDF}(K, \alpha)$ for a given K, α, and set S_{train} using the multi-model technique (Alg. 1):

Algorithm 1 Computing $P_{CDF}(K, \alpha)$

Input: S_{train}, α, K
Output: $P_{CDF}(K, \alpha)$

```
Define models M = {1, ..., M} using S^train.
Compute τ_ij(m), ∀m ∈ M.
for E = 0 to K do
    Calculate P_T(E, α) for a given α;
    Switch to ξ, min_{ξ∈M} |τ_{i,j}(ξ) − t_ij| with each event
end for
Calculate P_CDF(K, α) as in Equation 6.
return P_CDF(K, α)
```

Initially, given a set S_{train}, the predetermined set of models $\mathcal{M}$ is formulated as in Section III-A and the mean of each model is computed. Thereafter, the probability of a particular number of events $P_T(E, \alpha)$, for $E = [0, K]$, is calculated as follows: each time an out-of-bound event occurs, operating model ξ is changed to the model with a mean travel time, $\tau_{i,j}(m)$, closer to the triggered travel time violation. Finally, $P_{CDF}(K, \alpha)$ is calculated as in Equation 6.

The objective of Algorithm 1 is to establish the relationship between P_{CDF}, and parameters K and α. Consequently, for any given value of the violation bound, α, we can compute the probability of achieving the desired number of events K. Figure 2 illustrates this relationship as the probability P_{CDF} is calculated for various parameters K and α for a 3-model case. For instance, the probability to have no more than $K = 12$ events using a violation bound $\alpha = 220$ seconds is $P_{CDF}(12, 220) = 0.92$. Thus, for the case of a testing set of 100 trips, 92 should produce no more than 12 events while the rest should produce more than 12.

The established relationship between $P_{CDF}(K, a)$ and the parameters K and α allows the calculation of P_{CDF} for any target number of events $K \in [0, \mathcal{B} - 1]$ and any percentage of the target K, defined as $p_{target} \in [0, 1]$, by tuning accordingly the magnitude of bound, α. Consequently, we define the tuning parameter α for the given K and p_{target} as the desired bound $\alpha_d(K, p_{target})$. The parameter α_d is the minimum value of α that drives $P_{CDF}(K, \alpha)$ to become p_{target} for a given K. This can be mathematically formulated as follows:

$$\alpha_d(K, P_{CDF}) = min(\alpha) \mid P_{CDF}\big(K, \alpha\big) = p_{target} \quad (7)$$

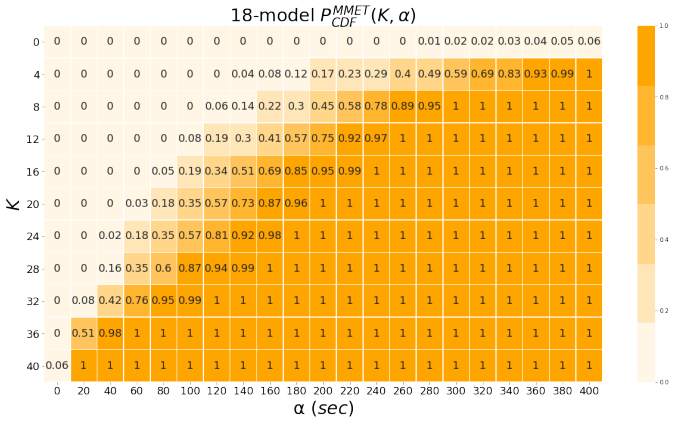

Fig. 2. $P_{CDF}(K,\alpha)$ for various values of K and α for a 3-model MMET scenario.

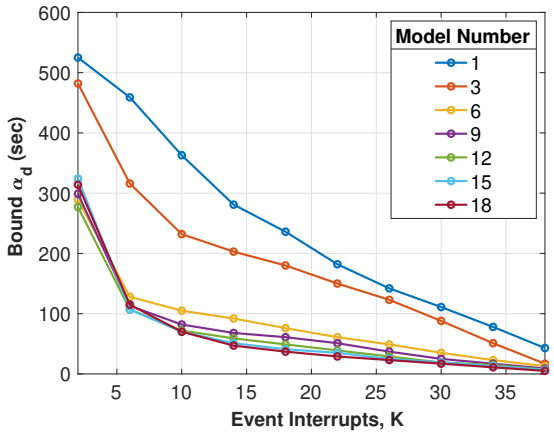

Fig. 3. Bounds for increasing $K \in [2, 38]$

Numerical methods such as the bisection method can be implemented to calculate the value of α_d. Algorithm 2 demonstrates the application of the bisection method for the calculation of α_d.

Algorithm 2 Calculating bounds for a desired K and p_{target}

Input: α_l, α_u, ϵ, K, p_{target}
Output: α_d

```
while α_u − α_l > ε do
    Compute α = (α_l + α_u)/2
    Compute P_CDF(K, α) using Algorithm 1
    if (P_CDF(K, α) > p_target) then
        α_u = α
    else
        α_l = α
    end if
end while
α_d=min(α)
return α_d
```

Initially, α_d is assumed to lie within the interval (α_l, α_u). We compare $P_{CDF}(K,\alpha)$ of $\alpha = \frac{\alpha_l+\alpha_u}{2}$ with p_{target} and update the interval accordingly. The process repeats until the predetermined precision ϵ is reached $(\alpha_u - \alpha_l > \epsilon)$.

To demonstrate the applicability of the technique, Algorithms 1 and 2 are applied to set $\mathcal{S}_{train}$ to calculate α_d for the case of one or multiple models. Figure 3 shows the computed bounds using the new algorithm for several values of K and $p_{target} = 0.8$.

As shown in the figure, the greater the number of models, the smaller the number of bound violations. Importantly, by observing the difference of the single-model bounds with those of the 18-model case, for $K = 10$, in the first case $a_d = 360$ and in the second $a_d = 80$; resulting to a 4 times reduction in the magnitude of the bound. Clearly, smaller bounds are preferable as a higher precision of the violation leads to better tracking accuracy. As expected though, an increasing number of models provides diminishing returns. For instance, the bounds for the cases of 12, 15 and 18 models are very similar.

The performance of the calculated bounds, α_d, are then evaluated on the test set S_{test} for any given K, α, and $\mathcal{M}$. A suitable error measure is chosen based on the absolute difference between p_{target} and the achieved $P_{CDF}(K,\alpha_d)$ as described by the loss function, L_f below:

$$L_f = mean\left\{\left\{\left|P_{CDF}(K,\alpha_d) - p_{target}\right|\right\}_{K=0}^{\mathcal{B}-1}\right\}_{\mathcal{M}=1}^{40} \quad (8)$$

Indicatively, the value of the loss function is found to be 12×10^{-3} for the case of the particular route that consists of $B = 41$ sections, and for $p_{target} = 0.98$, indicating that the proposed MMET technique can successfully relate the desired number of events with tight out-of-bound interrupts.

IV. Baseline Multi-Model ET

The aim of the Baseline MMET (BMMET) is to utilize the most likely behavior models of operation in an attempt to reduce communication between the two entities. As before, we consider an ET technique where multiple behavior models are generated and, whenever deemed necessary, switching between these models is performed. In this section, we will explore a switching-free MMET technique to track vehicles in order to minimize computational effort by the remote and local hosts.

A. Model Behavior

The baseline approach considers no adaptive switching and simply uses a predetermined model sequence for monitoring by the remote hosts. The predetermined set Ξ is defined as $\Xi = \{\xi_{ij}(m), \forall(i,j) \in \mathcal{B}, m \in \mathcal{M}\}$ where the most-probable models are used in recurrent service instantiations.

To aid understanding, Fig. 4 shows the distribution of models for a single road segment, $S_{1,4}$, while Fig. 5 shows boxplots of the model distributions across a particular route, Route 150, assuming that 18 models have been generated for each segment. Evidently, the distribution of models in each segment is not uniform (as shown in 4) and that distribution varies along the route segments substantially (as shown in

Fig. 5). For our baseline approach, we assume that the most frequent operating model for each segment along the route is used to characterize the vehicle mobility.

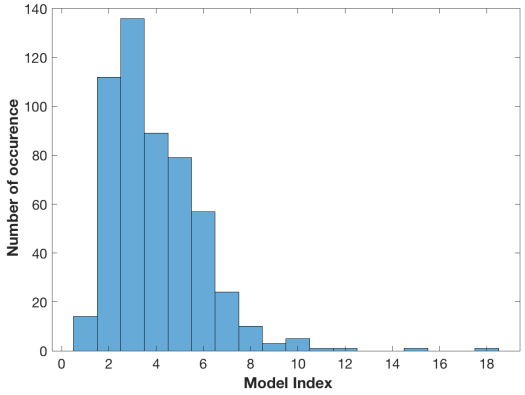

Fig. 4. Distribution of models for bus trip segment $S_{1,4}$.

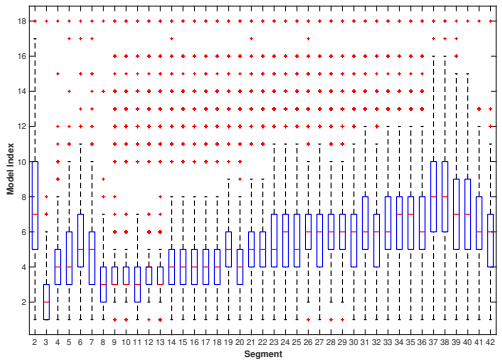

Fig. 5. Trajectory of model distributions for Route 150.

B. Event Detection and Event Handling

Since no switching between models is required, the event detection and event handling sections are similar to the one used for the single-model as described in [14]. The sole difference appears on the violation bounds, α. In this case, they are assigned around the mean of the predetermined operating model, $\alpha(\xi_{ij}(m))$, for each segment of the bus route.

V. Proactive Multi-Model ET

An extension of the MMET technique is considered hereafter where upon an event interrupt, the operating model switches to the one closer to the triggered travel time violation. The updated model sequence is then used by subsequent vehicles that operate in that particular route. Doing so could provide improvements in tracking performance and potentially minimize incipient changes.

A. Model Behavior

Whenever an event interrupt occurs, the local host informs of the event and both the local and remote hosts switch to the model closer to the triggered travel time violation. The updated operating model sequence for each segment of the route is subsequently used by other vehicles in order to better anticipate patterns and avoid unnecessary triggering of resynchronization events.

Similar to the case of the baseline MMET approach, the proactive MMET (PMMET) technique utilizes a mutual set of operating models denoted as $\Xi = \{\xi_{ij}(m), \forall(i,j) \in \mathcal{B}, m \in \mathcal{M}\}$. Unlike the baseline approach, set Ξ is updated online to maintain the most recent configuration of operating models used by local and remote hosts for synchronization and monitoring.

B. Event Detection and Event Handling

Each time a local host requests an update of the operating model for a particular segment, ξ_{ij}, set Ξ is updated and the updated set is used by all subsequent vehicles. In this way, the next vehicle to traverse the particular segment of the route will use the most recently updated model for monitoring and so will the remote host, in an effort to further minimize resynchronization events. Local hosts maintain an updated set Ξ by continuously adapting the modelling sequence for subsequent instances. To summarize, the following procedure is followed:

1) All hosts (both local and remote) operate on the models defined by the mutual set Ξ.
2) Whenever an event is generated by any local host, both local and remote hosts update Ξ.
3) Subsequent vehicles use the updated Ξ to monitor the mobility progress along the route.

VI. Experimental Results

To investigate the performance of the proposed technique, the different schemes described above are evaluated for different K and M values. For all the results considered hereafter all road segments are characterized by 18 models. The initial MMET scheme assumes that whenever an event is triggered, vehicles switch to the model with the closest mean but subsequent service instantiations ignore this switching. The baseline scheme (BMMET) also assumes that for each consecutive iteration, vehicles do not consider past experiences and, in addition, no switching takes place upon an event interrupt. Instead, vehicles always use the most frequent model extracted from historical data obtained for monitoring purposes. On the other hand, the PMMET scheme updates the model sequence for all subsequent instantiations based of the latest observations made for the specific route.

A comparison of the three schemes for various values of K and α are shown in Fig. 6. Clearly, the performance of the switching-capable MMET scheme is significantly better than BMMET, where the reduction in event interrupts from both the local and remote hosts for the baseline approach leads to looser bounds resulting in diminished tracking performance of the system. On the other hand, PMMET manages to improve its performance compared to the initial MMET scheme. As shown in the figure, the memory of past instantiations in PMMET manages to reduce the model bounds for smaller number of targeted events K ($K \leq 8$).

To further investigate this improvement, Fig. 7 plots the P_{CDF} of the MMET and PMMET schemes for $K \leq 8$

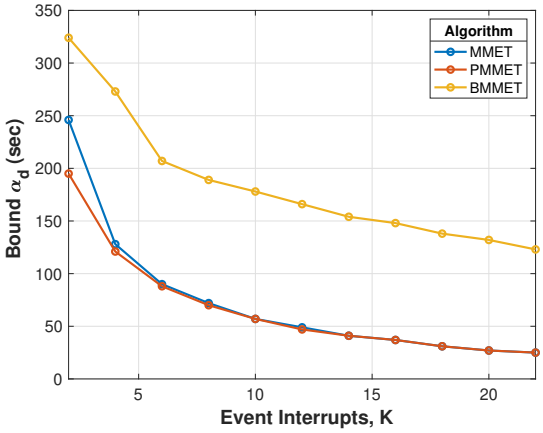

Fig. 6. Comparison of desired bounds, a_d, for $p_{target} = 0.7$ and $K \in [2, 22]$ for the 18-model MMET, Baseline MMET, and proactive MMET schemes.

and $\alpha \in [0, 800]$ as well as the difference between the two probabilities, $\Delta P_{CDF}(K, \alpha) = P_{CDF}^{PMMET}(K, \alpha) - P_{CDF}^{MMET}(K, \alpha)$. We can observe that $\Delta P_{CDF}(K, \alpha)$ is always positive or zero, meaning that PMMET improves or matches the performance of MMET. Furthermore, for smaller values of K, both the magnitude of $\Delta P_{CDF}(K, \alpha)$ as well as the number of non-zero $\Delta P_{CDF}(K, \alpha)$ values across the range of α is larger. This leads to tighter bounds and supports the observations from Fig. 6.

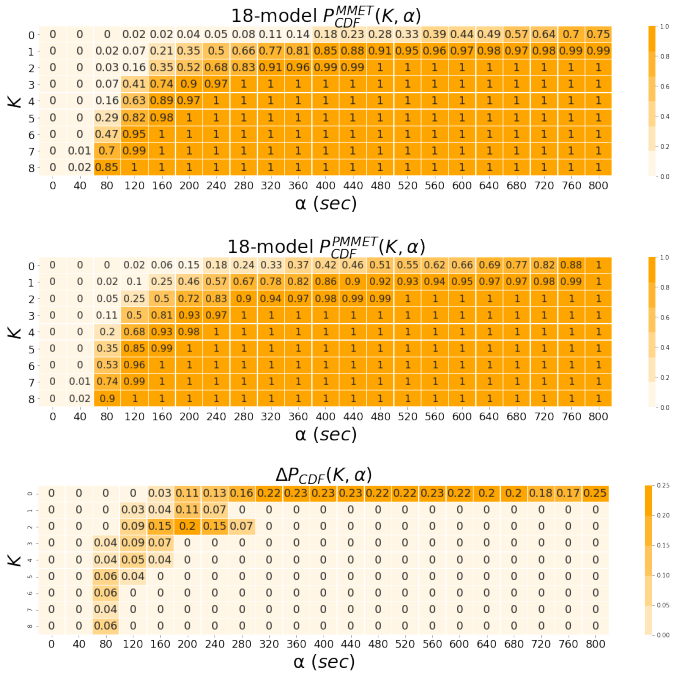

Fig. 7. *Upper:* $P_{CDF}(K, \alpha)$ of 18-model MMET scheme. *Centre:* $P_{CDF}(K, \alpha)$ of 18-model PMMET technique. *Lower:* Difference of $P_{CDF}(K, \alpha)$

VII. Conclusions

In this paper, the multi-model event triggering framework and two extensions of it have been developed that aim to improve the performance of tracking vehicles with recurrent mobility patterns, while reducing triggering events and thus minimizing energy consumption and communication bandwidth. The BMMET scheme aims at eliminating communication between the on-board and off-board entities and utilizes the multi-model behavior of the route to predict mobility and better track the vehicles. However, it is shown that the tracking performance of this scheme is sufficiently degraded compared to the basic MMET approach. On the other hand, the PMMET technique aims at improving the tracking performance of the system by introducing a memory into the system to enable the adaptive use of the correct operating model by the vehicles. As demonstrated using real data from public transportation services, PMMET significantly improves the tracking accuracy using a very small number of triggering events.

Acknowledgments

This work has been supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 739551(KIOS CoE) and from the Government of the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development.

References

[1] CompTIA, "Sizing up the internet of things." Research Brief, 2015.
[2] S. Taylor, "How service providers can help businesses to realize the promise of the IoT revolution." CISCO Report, 2016.
[3] A. Haroon, et al, "Constraints in the IoT: The world in 2020 and beyond." *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 11, 2016.
[4] A. Elmangoush, H. Coskun, S. Wahle, and T. Magedanz, "Design aspects for a reference M2M communication platform for smart cities." *Proc. 9th International Conference on Innovations in Information Technology (IIT)*, pp. 204209, 2013.
[5] A. Musa, J. Biagioni, J. Eriksson, "Trading off accuracy, timeliness, and uplink usage in online GPS tracking." *IEEE Transactions on Mobile Computing*, vol. 15, no. 8, Aug. 2016.
[6] D. Ashbrook and T. Starner, "Using GPS to learn significant locations and predict movement across multiple users." *Pers. Ubiquitous Comput.*, vol. 7, no. 5, pp. 275-286, 2003.
[7] Y. Ohsawa, K. Fujino, H. Htoo, A. T. Hlaing, N. Sonehara, "Real-time monitoring of moving objects using frequently used routes." *Proc. 16th Int. Conf. Database Syst. Adv. Appl.*, pp. 119-133, 2011.
[8] W. Heemels, K.H. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control." *Proc. IEEE Conference on Decision and Control*, Dec. 2012.
[9] O. Vermesan, and P. Freiss, "Internet of things - Converging technologies for smart environments and integrated ecosystems." River Publishers Series in Communications, 2014.
[10] S. Kartakis, A. Fu, M. Mazo and J. A. McCann, "Communication schemes for centralized and decentralized event-triggered control systems." *IEEE Transactions on Control Systems Technology*, vol. 26, no. 6, Nov. 2018.
[11] P. Huang, C. Wang, and L. Xiao, "RC-MAC: A Receiver-Centric MAC protocol for event-driven wireless sensor networks." *IEEE Transactions on Computers*, vol. 64, no. 4, Apr. 2015.
[12] P. Panagi and M. Polycarpou, "A coordinated communication scheme for distributed fault tolerant control." *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, Feb. 2013.
[13] T. Schlage and J. Lunze, "Modelling of networked systems for remote diagnosis." *Proc. Conference on Control and Fault Tolerant Systems*, Nice, France, October 2010.
[14] P. Kolios, C. Panayiotou, G. Ellinas and M. Polycarpou, "Data-driven event triggering forIoT applications." *IEEE Internet of Things Journal*, vol. 3, no. 6, Dec. 2016.