MODIFIED DIJKSTRA'S ALGORITHM WITH CROSS-LAYER QOS

Andrei B. RUS Virgil DOBROTA Adrian VEDINAS Georgeta BOANEA Melinda BARABAS
Technical University of Cluj-Napoca, Communications Department, 26-28 George Baritiu Street, 400027 Cluj-Napoca, Romania, Tel: +40-264-401226, E-mails: {bogdan.rus, virgil.dobrota, georgeta.boanea, melinda.barabas}@com.utcluj.ro, adrian_vedinas@yahoo.com}

Abstract: This paper presents a modified Dijkstra's algorithm that calculates the distance between multiple sources and a single destination. It corrects the deficiencies of the classical approach by taking into account the dynamicity of the QoS parameters at the Physical Layer and MAC Sub-layer. The proposed composite metric is based on the available transfer rate, one-way delay and bit error rate, all of them measured or calculated in real time due to a Cross-Layer QoS software module. The proof-of-concept was obtained by simulations in OMNET++.

Keywords: *Cross-layer, modified Dijkstra, QoS parameters*

I. INTRODUCTION

The routing protocols are an important factor that influences the quality of the services perceived by a user [1],[2]. Unfortunately majority of them are using static information to choose the best path between multiple sources and a given destination node. The legacy routing protocols are aware about the theoretical capacity of the communication channel, given by the technology used at hardware level. On the other hand, to take optimal routing decisions, more than one parameter at the lowest level possible should be considered. Besides the available transfer rate (ATR) of the communication link, we are proposing in this paper, whenever choosing the best path, to consider also the one-way delay (OWD) and the bit error rate (BER).

Furthermore, another issue addressed herein is related to real-time measurements of previously mentioned QoS parameters. Thus, if the values are not updates, a link with a transfer rate theoretically higher than of the others, but encountering congestion, could have actually worse performances. Therefore, we are proposing a set of key performance indicators monitored through real-time passive or active measurements on top of MAC Sub-Layer. Based on these parameters the spanning tree of the considered network will be constantly updated, adapting it to any changes that may appear. Due to cross-layer techniques involving a local database, information from lower layers is offered to Network Layer routing processes (e.g. OSPF – Open Shortest Path First).

For optimal decisions with respect to the shortest path to a given destination within modified Dijkstra's algorithm, each node should be aware of the actual performance of all communication links in the network. Furthermore, in order to achieve a certain level of situation awareness in each device, the measurements performed by other peer nodes are published within that routing domain.

The paper is structured as follows: Section II describes briefly the classical Dijkstra's algorithm with its drawbacks mentioned above. The third section presents in details how to modify the existing algorithm to make it CLQ (Cross-Layer QoS)-aware. The Section IV evaluates the performances of the modified Dijkstra's algorithm, through simulations performed in OMNET++ [3], [4]. The last section concludes and suggests issues for future work.

II. OVERVIEW OF THE DIJKSTRA'S ALGORITHM

Suppose the multiple sources and a single destination version of Dijkstra's algorithm. The main goal is to find the spanning tree that includes the shortest path (with respect to the minimum cost) from each node in the topology to a given destination. We used the following notations: D_i is the cost of the path from the source node *i* to the destination node *d*; d_{ij} represents the distance of the link between nodes *i* and *j*. To simplify the testing, we calculate a fixed minimum cost function $D_{ij} = d_{ij} \geq 0$, as within OSPF protocol. The formula for the single metric is the following:

$$D_i = D_{ij} + D_j = \frac{10^9 [bps]}{C_{ij}[bps]} + D_j \quad (1)$$

where: $C_{ij} \geq 0$ is the theoretical link capacity at Network Layer between node *i* and node *j*. Suppose P is the set of nodes for which the shortest path was detected, whilst Q is the set of nodes for which the shortest path has not been detected yet. Observe that in this protocol both the distances and costs are non-negative numbers. The steps of the Dijkstra's algorithm, described in [6], are the following:

Manuscript received September 9, 2010; Revised November 3, 2010

Initially, the only one node included within the set *P* is the destination, while the rest of them are still in the set Q. Moreover, the cost of the path from the destination node to itself is always equal to zero. All the others have the initial values equal to the distance of the link between nodes *i* and destination node *d*:

$$P=\{d\}, Q=\{1,2,..j\}, D_d=0, D_j=d_{jd}, \forall j\neq d \qquad (2)$$

Step 1:: The next closest node to the destination is detected, then it is included within the set P. Note that the verification is applied exclusively to the nodes that are within the set Q.

$$D_i=\min_{j\notin P} D_j, \forall i\notin P$$
$$P=P\cup\{i\} \qquad (3)$$

If all the nodes are contained by the set *P*, the algorithm will stop, otherwise it will continue with the next step.
Step 2: The costs of the routes to the destination from all the nodes within the set Q will be updated. The algorithm chooses the path characterized by the minimum value of the cost from all available alternatives.

$$D_j=\min_{i\in P, j\notin P}[D_j, d_{ji}+D_i] \qquad (4)$$

After the updating of all distances, the algorithm returns to *Step 1*. As a major drawback, Dijkstra's cost refers to Network Layer only, without seizing for instance the congestion that may occur. Thus the latency, i.e. one-way delay for the link between *i* and *j*, could be significantly higher than the distance d_{ij}, if the average waiting time within the queue (service time not included) TW_{ij} is not negligible.

$$OWD_{ij}=d_{ij}+TW_{ij} \qquad (5)$$

III. MODIFIED DIJKSTRA'S ALGORITHM WITH CROSS-LAYER QOS

In this section we will present in details how Dijkstra's algorithm has been modified in order to use the services offered by the cross-layer QoS module.

When choosing the best path available, from multiple sources to a single destination, a set of key performance indicators KPIs have to be monitored in real-time. They characterize objectively the performances of the communication channel (i.e. total available transfer rate, total one-way delay and total bit-error rate). The indicators were aggregated into a composite metric (CM) [7]

$$CM=\frac{K_0}{ATR_T}+\frac{OWD_T}{K_1}+K_2\times BER_T \qquad (6)$$

where $K_0=10^9[bps], K_1=10^{-5}[s], K_2=10^{12}$. Note that index T refers to the total value of the parameter for the complete path. The *K* constants were chosen to allow the composite metric CM to have a minimum value of 3 for a link with the capacity 1 Gbps, one way delay 10 μs and a bit error rate equal to 10^{-12}. The second modification brought to the Dijkstra's algorithm is related to the way that the composite metric is computed for an entire path. In the classical approach, equation (3) is used to calculate the cost of a route. Thus, the sum of all the distances characterising the links of the analysed route is considered. Note that in our proposal the cost from *i* to *d*, i.e. $D_i, \forall i\in P$, is actually the composite metric CM of that path.

The ATR_T, OWD_T and BER_T parameters are calculated globally for all the concatenated links of a specific path. However their composability is different. Thus the total available transfer rate of a path is equal to the minimum ATR of all the links that are composing it.

$$ATR_T=\min_{i,j}(ATR_{ij}) \qquad (7)$$

where ATR_{ij} represents the available transfer rate of the link between nodes *i* and *j*. On the other hand the total one-way delay of a path is equal to the sum of the OWDs of all the links that are composing it.

$$OWD_T=\sum_{i,j} OWD_{ij} \qquad (8)$$

where OWD_{ij} is the one-way delay of a link between nodes *i* and *j*. Finally the total bit-error rate of the path is considering the BERs of each link that is composing it.

$$BER_T=1-P_T \qquad (9)$$

$$P_T=\prod_{i,j}(1-BER_{ij}) \qquad (10)$$

where BER_{ij} represents the bit-error rate of a link between nodes *i* and *j*, whilst P_T is the probability of not having erronated bits on the whole path.

Note that there are several changes in the modified routing scheme proposed herein, besides the replacement of the simple metric given in equation (1) by the composite metric from equation (6). Thus the classical Dijkstra's algorithm stops when the spanning-tree of the shortest path is obtained (i.e. all the nodes are included into the set P). Unless major changes in the link occur (failures, new links available or new technologies at Data Link Layer) the algorithm is not restarted. However the modified Dijkstra's algorithm is running continuously, because at least the monitoring part of ATR, OWD and BER is always activated due to Cross-Layer QoS. This paper offers a proof-of-concept only and it is beyond its scope to evaluate the optimization issues like control overhead, route oscillation

and others. The classical algorithm is treated as a particular case of modified Dijkstra's if the following conditions are fulfilled: a) $K_1 \to \infty, K_2 = 0$; b) ATR is always equal to the link capacity; c) Cross-Layer QoS is not activated.

IV. TESTING THE MODIFIED DIJKSTRA'S ALGORITHM WITH CROSS-LAYER QOS

For demonstrating the benefits of adding the Cross-Layer QoS mechanism, a testbed was simulated in OMNET++ as in Figure 1.

A number of 13 routers were included into the tested topology to ensure that enough multiple paths are available between multiple sources and a destination node. The network is abstracted by a graph with oriented edges, thus a full-duplex connection is represented by two oriented links. The traffic transmitted through the network will travel from the source node H0 to the sink node H12.

A. Varying the Available Transfer Rate

The main purpose of this test is to verify how the Dijkstra's algorithm together with the CLQ seizes the dynamic changes of the available transfer rate on different links in the network.

The results will be compared with the ones obtained when employing the classical Dijkstra's, on the same topology and in the same conditions. To be able to make a comparison between the two solutions, the end-to-end delay parameter of a video flow will be monitored.

Based on the spanning-tree computed with both algorithms, the routing protocol (i.e. OSPF) can detect the shortest path from any source node to a destination. Moreover the routes are being added into the routing table of each node in the network. The testing scenario includes two simulations (one for each algorithm). In both of them, the performances of the two links were changed and then the results were analyzed.

- *Step 1:* ATRs of all links were set to a default value ≥ 1 Mbps
- *Step 2:* ATR of the link R0-R4 changes from 5 Mbps to 500 kbps
- *Step 3:* ATR of the link R4-R7 changes from 3 Mbps to 300 kbps

The influence on the routing protocol will be presented in the following figures. The continuous line marks the path taken by the packets when simulating the modified Dijkstra, while the dotted one shows the path between H0 and H12 within the classical algorithm.

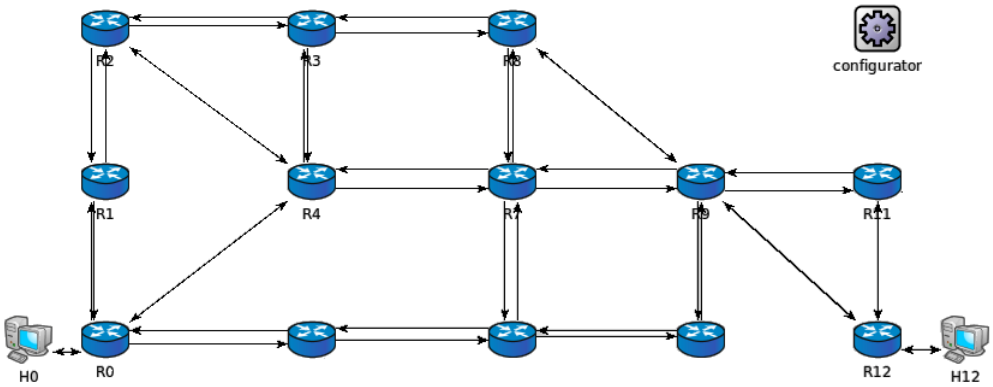

Figure 1. Testbed used to simulate Dijkstra with Cross-Layer QoS

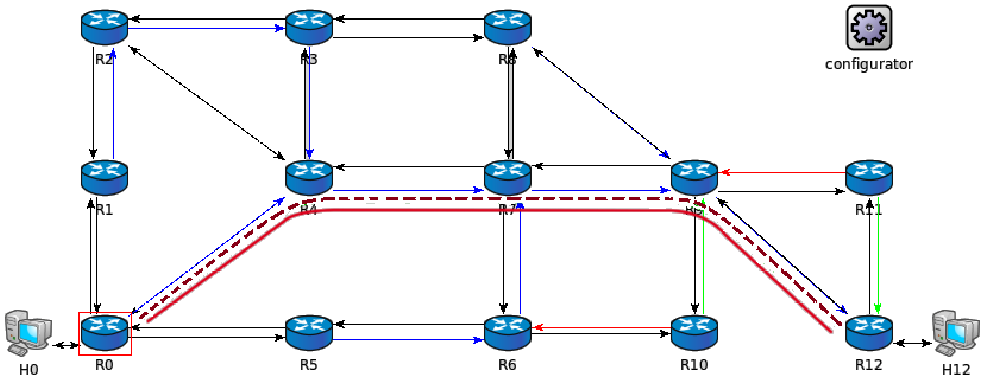

Figure 2. Path between H0 and H12 in Step 1

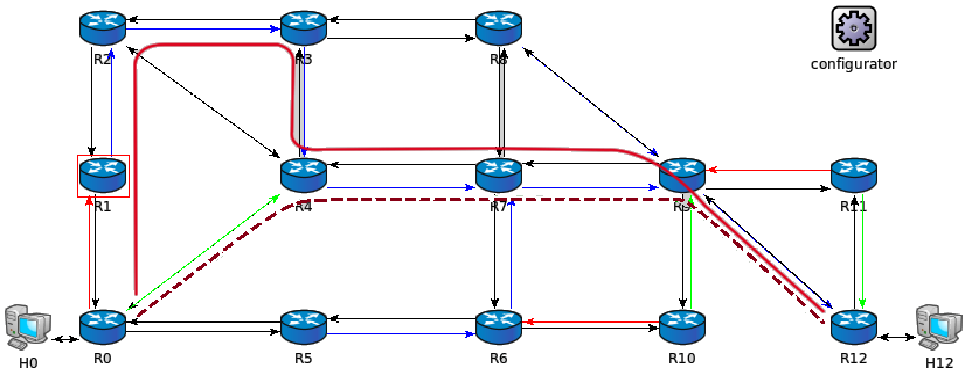

Figure3. Path between H0 and H12 in Step 2

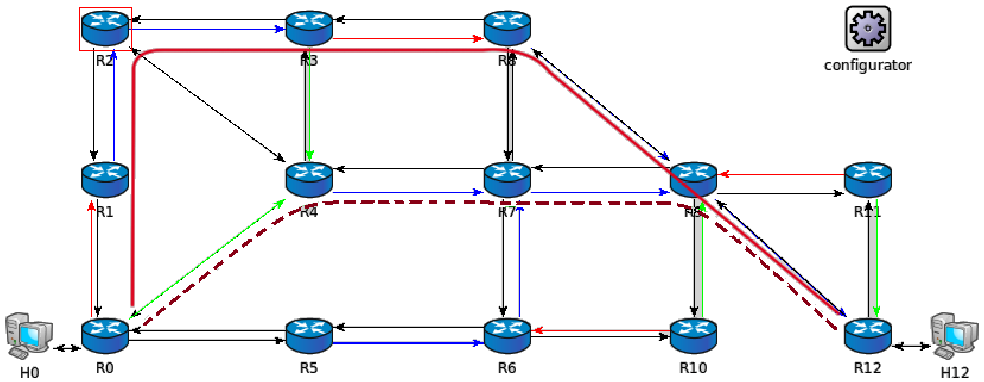

Figure 4. Path between H0 and H12 in Step 3

If comparing the Figures 3 and 4, one can observe that the variation of the network performance is transparent to the classical algorithm. This is due to the fact that the forwarding path between the source node H0 and the destination one H12 does not change at all during the three steps simulated.

When activating the modified Dijkstra's algorithm, the spanning-tree and the path between H0 and H12 are changing, every time when a link parameter is modified. Thus the best available routing solution could be used and the video stream transmitted experienced a better quality with respect to the end-to-end delay (EED), comparing it to the classical approach (see Figure 5).

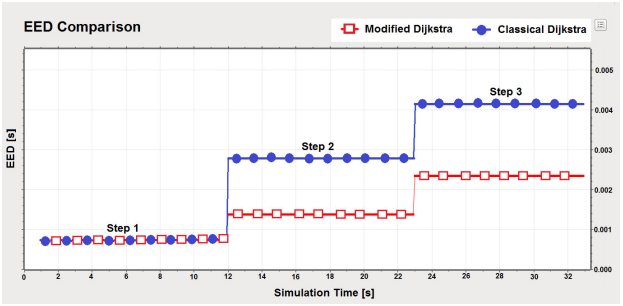

Figure 5. End-to-end delay comparison when ATR is variable

Numerically, the average value of the EED was equal to 0.0025 [s] for classical algorithm and 0.00145 [s] for the modified version.

B. Varying the One-Way Delay Parameter

To measure the performances of the proposed approach, a video flow was sent again from the source H0 to the destination H12. Similar with the previous case, the simulated scenario included three steps:

- *Step 1:* OWDs of all links were set to a default value < 10^{-3} [s]
- *Step 2:* OWD of the link R0-R4 changes from $2*10^{-4}$ [s] to $2*10^{-2}$ [s]
- *Step 3:* OWD of the link R4-R7 changes from $2*10^{-4}$ [s] to $2*10^{-2}$ [s]

The first set of simulations used the classical Dijkstra's algorithm while the second one employed the QoS-aware modified version. The video flow stream uses first the path H0-R0-R4-R7-R9-R12-H12.

Within the second step, the quality of the link R0-R4 drops down influencing the composite metric accordingly (i.e. the value will increase). Thus when the spanning tree of the topology is recomputed, the link R0-R4 from the initial route is changed automatically with the section R0-R1-R2-R3-R4 because of its better performance. When simulating the third step, due to the fact that the OWD of the link R4-R7 increased, another change is made with respect to the route

between H0 and H12. This consists of changing the section R0-R4-R7 with R0-R2-R3-R8-R7. Thus the EED parameter of the video flow is kept to a lower value. Comparing the averyage value of EED in both simulations, one can observe that the classical version of Dijkstra's algorithm performed worse (i.e. 0.0021 [s]) than the modified one (i.e. 0.0005 [s]). The evolution of the instant values of EED parameter can be seen in Figure 6.

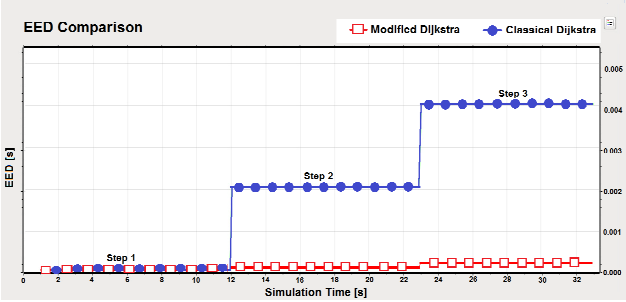

Figure 6. End-to-end delay comparison when OWD is variable

C. Varying the Bit Error Rate Parameter

In this case, the topology remained the same as in the previous ones, with the following three steps:

- *Step 1:* BER parameter of all links is set to a default value of around 10^{-11}
- *Step 2:* BER of the link R0-R4 changes from $6*10^{-11}$ to $6*10^{-5}$
- *Step 3:* additionally the BER of the link R4-R7 changes from 10^{-11} to 10^{-5}

For the classical Dijkstra's, the path between H0 and H12 remained the same (i.e. H0-R0-R4-R7-R9-R12-H12) during the three steps of the test. When the modified Dijkstra's algorithm was employed, the link R0-R4 was changed with the section R0-R1-R2-R3-R4 in the second step. During simulations within the third step, the section R0-R4-R7-R9 was changed with R0-R1-R2-R3-R8-R9. When simulating the classical Dijkstra, because the BER parameter varies and the paths with higher error probability cannot be avoided, from the total of 320 packets sent, only 242 were received at the destination node. With the modified algorithm simulated, all the packets sent were received but because the path was changed to one characterised by a higher value of OWD, the EED parameter increased accordingly (see Figure 7). The average value of the EED parameter measured on the video stream was 0.0007 [s] for the classical Dijkstra and 0.0014 [s] for the modified version of the algorithm. We consider that the increased value of the EED parameter is an acceptable trade-off for keeping the percentage of video packets lost as low as possible. If a certain application would prefer the path with the minimum delay, even if there is a higher value of the BER, the coefficient K_2 from equation (6) can be tuned so that the error probability will have less influence on the composite metric.

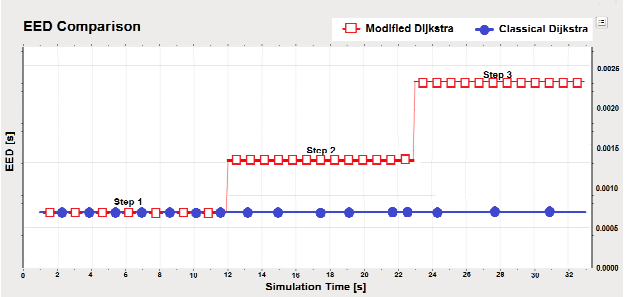

Figure 7. End-to-end delay comparison when BER is variable

D. Varying ATR, OWD and BER Parameters

In the last set of experiments, all three parameters (ATR, OWD and BER) were modified simultaneously in three steps to the same values as in the previous scenarios. The results of the simulations can be observed in Figure 8.

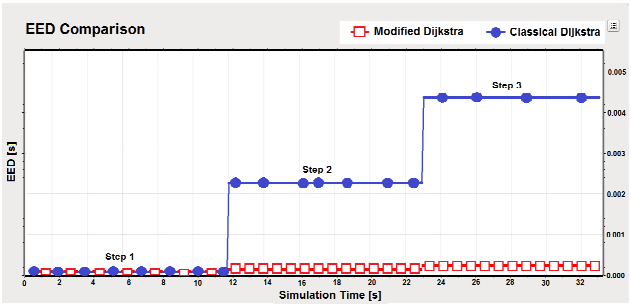

Figure 8. . End-to-end delay comparison when ATR, OWD and BER are variables

During the first step the route followed by the video stream from H0 to H12 was the same (i.e. H0-R0-R4-R7-R9-R12-H12), no matter the algorithm employed.

Within the second step, the parameters of the link R0-R4 were degraded, so the route between H0 and H12 used the section R0-R1-R2-R3-R4 instead. The path changed because the spanning tree of the topology was modified when the composite metric of all the routes were recomputed (using the values gathered in real-time from the cross-layer QoS).

The third step decreased in addition the performance of the link R4-R7 (with respect to the ATR, OWD and BER parameters) resulting another change of the path between H0 and H12. Thus, the section R0-R4-R7-R9 changed to R0-R1-R2-R3-R8-R9.

In Figure 8, the advantages of using the modified Dijkstra's algorithm with cross-layer QoS are obvious. One can see that the classical algorithm does not react to performance changes in the network even if the path's quality decreases. Besides the fact that in the classical approach the EED parameter is higher than in the quality of service-aware one, the video flow is affected additionally by lost packets.

V. CONCLUSIONS

In this paper a modified Dijkstra's algorithm that uses real-time QoS information from the lower layers is proposed. The information is offered to the Network Layer where the routing algorithm is running, using a cross-layer bottom-up signalling. The second contribution presented herein is the composite metric formula used to calculate the cost of a path from multiple source nodes to a destination. The cost is recalculated periodically and similar for the spanning tree. Thus, if the quality of a route decreases, the section with problems will be avoided (if an alternative exists).

The results obtained during simulations are encouraging us to continue this preliminary research. The modified Dijkstra's algorithm provides an end-to-end delay (EED) about 10 times lower in the conditions of a link state deterioration (100 times OWD increases, ATR decreases of 10 times and a six orders of magnitude BER increases) on two network segments of the route. In the future, a lot more tests are envisaged in which a correlation between the parameters variations corresponding to real practical situations will be considered. We plan also to study the link oscillation phenomenon and ways of aggregating the QoS information in order to make the proposal completely scalable.

REFERENCES

[1] J.Macfarlane, "Understanding IP Routing in Cisco Systems", *Wiley Publishing*, 2006

[2] D.Medhi, K.Ramasamy, "Network Routing Algorithms, Protocols, and Architectures", *Morgan Kauffman Publishers*, 2007

[3] A.Varga, "OMNeT++ 4.0 User Manual", 2009, http://omnetpp.org/doc/omnetpp40/manual/usman.html

[4] A.Varga, INET Framework for OMNeT++/OMNEST, 2010 http://inet.omnetpp.org/doc/INET/neddoc/index.html

[5] ***, "OSPF Design Guide", *Cisco Systems*, August 2005, http://www.cisco.com/en/US/tech/tk365/technologies_white_paper09186a0080094e9e.shtml

[6] V.Dobrota, "Switching and Routing Systems", Technical University of Cluj-Napoca, 2010, http://el.el.obs.utcluj.ro/scr/.

[7] A.B.Rus, M.Barabas, G.Boanea, Zs.Kiss, Zs.Polgar, V.Dobrota, "Cross-Layer QoS and Its Application in Congestion Control", *17th IEEE Workshop on Local and Metropolitan Area Networks, LANMAN 2010*, Long Branch, NJ, USA, May 5-7, 2010