0.1 ACL optimization

Definition 0.1.
Let $d_i(L)$ be the full delay of demand on list L, which matches against the rule i in list L and before rule i it does not matches against any rule [?],[?].

$$d_i(L) = \sum_{k=1}^{i} r_k(L) \tag{1}$$

Definition 0.2.
Used the expression (1), let the overall latency on list L in case of traffic T be defined in (2), if there is no queuing [?],[?].

$$Delay(L,T) = \sum_{i=1}^{n(L)} m_i(L,T) \cdot d_i(L) = \sum_{i=1}^{n(L)} m_i(L,T) \cdot \sum_{k=1}^{i} r_k(L) \tag{2}$$

Be this ratio the weight of the specified rule i in the list L.

Definition 0.3.
Two list-rules depend from each other, if they have common parts, so there are demands that would match against both of them. Point the direction of the dependency into the first rule in the list from these two rules, according to (3) [?],[?].

$$A \Leftarrow B \text{ if } \exists C, \text{ that } C \subseteq A \text{ and } C \subseteq B \tag{3}$$

Definition 0.4.
A rule in the list is redundant, if it directly depends on an other rule and the direction of the dependency points into the redundant rule. This redundancy is true if and only if the dependency generator rule is not in indirect dependency with the redundant element, and the element is the full-subset of the dependency generator element, as it is described by (4) [?],[?].

$$A\Re \text{ if } A \Leftarrow C \text{ directly, and not} \exists B, \text{ that } A \Leftarrow B \Leftarrow C \tag{4}$$

Thesis 0.1.
Based on my definitions (Definition 0.1-0.4) I created a structure, which describes the dependencies and connections between the rules of the list, and minimizes the redundancy in the structure. This structure is based on directed graphs which also may contain cycles in it [?],[?].

Thesis 0.2.
I created algorithms, based on the constructed directed graph and weights (Thesis 0.1), which can be used to reduce the packet delay on the Access List. These algorithms are heuristics based on approximated traffic. I have chosen the best of them by experimental measurements according to execution time and efficiency [?],[?].

0.2 ACL Modeling

Thesis 0.3.
I created mathematical models (based on my model parameters in Table **??***), which can describe the ACL and other filter structures, and we can predict performance parameters from it. An infinite model is created (Figure* **??** *and Equations* **??***-***??***), which can be solved relatively fast, but it can only approximate the performance parameters. I also developed a finite model (Figure* **??** *and Equations* **??***-***??** *and* **??***), which is slower, but more accurate [***?***],[***?***].*

Thesis 0.4.
I defined an upper boundary (Equation **??** *and* **??***), under what the infinite model results (Thesis 0.3) can approximate well (below* 10^{-5}*) the results of the finite model, in case of any kind of matching probabilities in the ACL. The boundary is given in case of Poisson traffic [***?***].*

Thesis 0.5.
I created mathematical models, and I have given a description method in mathematical form, for general modeling of hierarchical list-structures. A mathematical model of the BSD filter system is created (Equations **??***,* **??** *and Figure* **??***), based on the notations of Table* **??***, and a general model to describe hierarchical-lists and TTCNv3 alternative behavior is also introduced with (***??***)-(***??***) and with Figure* **??***, based on Table* **??***. With the help of general Markovian methods performance parameters can be calculated from my models [***?***].*

0.3 Interface Modeling

Thesis 0.6.
I have developed a mathematical model (Figure ?? and Figure ??), based on the parameters in Table ??, which can describe the effects of the hardware interfaces of a router, together with the embedded system in it. This model can handle the preemptive packet handling, which use interrupts.

Thesis 0.7.
I successfully decreased the state-space of the Interface model (Thesis 0.6). I applied the assumption (5) and I created a one-dimensional model (Figure ??) from the two-dimensional one (Figure ??).

I applied my assumption (Equation 5) during decreasing the state-space of my model. Namely that during packet decoding at most one new packet arrival can happen into the input interface. This way the state space explosion can be avoided. From the two-dimensional structure we can create a one-dimensional model.

$$P_r(\#\ of\ arrivals\ during\ S_A > 1) \approx 0 \tag{5}$$

Thesis 0.8.
I propose the approximation Equation 6 for the calculation of the second part of (??), the overall delay, what a packet spend in the described system. The second part of (??) describes the mean value of the packet delay, when the examined packet is at the level i in the phase k.

$$E(delay|N = i, J = k) = E(T|N = i, J = k) + E(delay|N = i, J = k) \cdot T_{IN} \cdot D_1 \tag{6}$$

Thesis 0.9.
I propose a method and structure (Figure ??,Figure ?? and Equations ??-??) that is applicable to calculate the exact value of the second part of (??). This equation is used to assign the overall latency, what packets spend in the described system. This method can be generally used for DQBD structures that use preemptive behavior.

Thesis 0.10.
My given method (Figure ?? and Equation ??) reduce the state-space of the proposed model (Thesis 0.7) according to a new idea (Figure ??). I assumed that the derived performance parameters do not depend on the distribution of the input interface model-part. I assumed that the performance parameters (packet delay and loss) depend only on the mean value of this distribution. I have proved this assumption and I successfully decreased the state-space of the model.