Effective Combinating of Wormhole Deadlock-free Routs

Aleksandr Karandashev, Valentin Olenev
Saint Petersburg State University of Aerospace Instrumentation
Saint Petersburg, Russia
{aleksandr.karandashev, valentin.olenev}@guap.ru

Abstract — **Current paper discusses the problem of tracking of deadlock-free routes. A brief overview of existing software tools providing this functionality is given. A complete overview of the proposed software for building routes for given SpaceWire onboard networks is presented. The paper discusses the application of different existing methods for the choosing of the best route from the list of the deadlock-free routes. A brief overview of the methods for of choosing the best route according to the provided criteria is given. A new method for choosing of the best route and its modification is proposed. Authors provide the result of the methods application and the detailed comparison.**

I. Introduction

One of the most important tasks in implementation of next generation spacecraft and aircraft is the development of on-board data transmission networks [1]. To develop a high quality and reliable network, engineers need a specialized software tool that will automate routine work and draw the developer's attention to the other important design stages. Such software helps to design networks and carry out simulations for specific communication protocols. One of the most popular data transfer protocols for the space industry is SpaceWire. Developers of SpaceWire on-board networks also need a software that will allow to design and simulate the network, track the deadlock-free routes and provide the user with the prototype of network configuration. Such software should be user-friendly, reliable and automated. It should present information in a visual form to simplify the development process. Such software is SANDS [2]. It allows using four software components that cover main stages of the on-board networks design: topology design, deadlock-free routes tracking, generation of scheduling tables and simulation of the network prototype.

The paper discusses the problem of choosing of the best configuration for deadlock-free routes within the SANDS software component, which provides solution for this task. Solving this problem we used such research methods as analytical research, simulation modeling, testing and comparative analysis.

The paper begins with overview of various software tools for on-board networks design and analyzing each of them. The analysis shows the advantages of the SANDS software and its relevance for the task of choosing the route configuration for SpaceWire onboard networks [3]. Then paper provides a brief overview of the SANDS and the internal mechanisms for tracking of deadlock-free routes. Second section of the paper provides the discussion of the problem of choosing the configuration of deadlock-free routes and the requirements for the developed method. Third section of the paper related to finding a solution to the presented problem, several methods that allow to choose the best route are considered. The analysis summary is provided in a comparison table. As a result, we form a number of methods that solve the problem of choosing the best configuration of deadlock-free routes. Each of the generated methods is also assessed according to previously stated criteria and the best of the methods is selected. The next section provides a detailed description of the final method for choosing of the routes best configuration and its possible modifications that have been implemented in the SANDS software. After that, paper presents the test scenarios of on-board networks for verification of the implemented method. Evaluation of the testing results allows to summarize the work results and formulate the main achievements of the study.

II. Overview of existing solutions

A. Software for simulation of onboard networks

To automate the SpaceWire on-board networks design, a specialized software is required. The space industry requirements for such a tool is presented in [4]. Therefore, the tool should:

1) Be easy to use;
2) Visualize the on-board network topology;
3) Track the deadlock-free routes;
4) Be reliable;
5) Be automated;
6) Support the SpaceWire protocol.

In the papers [5] and [6], such software tools were considered in details. These tools were evaluated for suitability for use in the design of SpaceWire on-board networks. The following software tools were considered: RRAS services of routing and remote access, AFDX network design tool, Network simulator (NS-3), Software ElectriCS Pro 7 Aviation, Online application CAD5D, Microsoft Visio program, OMNeT ++.

Routing and Remote Access Service (RRAS) is the application programming interface and Microsoft server software. RRAS allows to create applications that provide routing in IPv4 and IPv6 networks. Developers can use RRAS to implement routing protocols.

AFDX network design tool defines limits on the latency, size, and transmission periods of messages. In addition, it takes into account the features of AFDX networks. Features such as the ability to use different buffering strategies on the switch's output ports, or the ability to send multiple messages over a single virtual channel.

The NS-3 simulator is a discrete event network simulator designed primarily for research and educational purposes. NS-3 is used for tracking of the routes. It uses a Dijkstra Shortest Path First (SPF) algorithm in the topology for each node. In addition, it supports simulation of TCP, UDP, ICMP, IPv4, multicast routing, P2P and CSMA protocols.

ElectriCS Pro Aviation is designed to create a complete digital model of on-board electrical equipment of aircraft, design schematic diagrams and wiring diagrams of electrical systems for the project technical documentation.

The CAD5D online software is used for designing of local computer networks of enterprises and administrative buildings. It can be used to develop detailed floor plans of a building, place the necessary subscriber points and main control nodes, and calculate the necessary lengths of all cables.

Microsoft Visio is a program that allows creation of graphs, drawings, diagrams, and flowcharts. The application helps to present graphical information in a simple and accessible way.

OMNeT++ is an extensible, modular C++ component-based simulation library designed for creating network simulators. The result of the comparative analysis is presented in Table I.

TABLE I. COMPARATIVE ANALYSIS OF SOFTWARE TOOLS

	Easy to use	Visualize topology	Deadlock-free routes	Automated	Support SpaceWire
RRAS Services	+	-	+	+	-
AFDX tool	+	+	-	+	-
Simulators NS-3	+	+	-	+	-
ElectriCS Pro 7	-	+	-	+	-
CAD5D	+	+	-	-	-
Microsoft Visio	+	+	-	-	-
OMNeT++	-	-	-	-	-
SANDS	+	+	+	+	+

Analysis shown that only the SANDS meets all the stated requirements. However, each software tool has its own advantages: one allows to model networks, the other has good graphical interface. All considered software meet the requirement of reliability. In some cases, several network design tools can be used in the same project, but this is not convenient, it complicates the development process.

B. *The SANDS software tool*

SANDS supports the full cycle of development and modeling of on-board networks, which begins with the automated generation of network topology and finishes with the obtaining of simulation results, statistics and various diagrams [2]. SANDS includes almost all the functionality required for prototyping of a real on-board network. Paper [7] provides the status of the SANDS tool implementation.

SANDS currently includes SpaceWire implementations and two transport layer protocols: RMAP [3] and STP-ISS [8].

The designed network is exported to an intermediate XML format to be used in other components for modeling, tracking of routes, or generation of scheduling tables. The graphical interface displays the simulation results on network structure, graphs and diagrams.

SANDS has four main software components:

1) Component for the network physical structure design and evaluation of the physical characteristics;
2) Component for tracking of deadlock-free routes;
3) Component for the generation of scheduling tables for STP-ISS transport protocol (for data transmission with scheduling quality of service) [9];
4) A network simulation component that processes all the data from the previous three components and the graphical user interface [10].

Component № 2 is responsible for tracking of deadlock-free routes in the network [11]. It is one of the most important tasks in network design, especially for SpaceWire and its wormhole routing. The task consists in finding the sequence of switches between the transmitter and receiver and creating routing tables for those switches. Configuring routing tables in switches enables static routing for the SpaceWire network. In addition, the routing component provides the ability to build redundant routes. In addition, it also calculates the worst-case latency for transmission of data and SpaceWire control codes.

C. *Up/Down Routing Algorithm*

The base algorithm of the SANDS second component is the Up/Down Routing algorithm. Its detailed description is provided in [7]. The algorithm is based on the “up / down” rule: the correct route firstly shall cross zero or more links in the “up” direction, and then zero or more links in the “down” direction. In this way, channel dependencies are avoided since the packet cannot cross the link in the "up" direction after the cross in the "down" direction. To solve the problem of deadlocked routes, it is necessary to remove the cycles in the directed graph of the network topology. Up/Down Routing algorithm solves this task [12]. The process of deadlock-free routes obtaining is illustrated in Fig. 1.

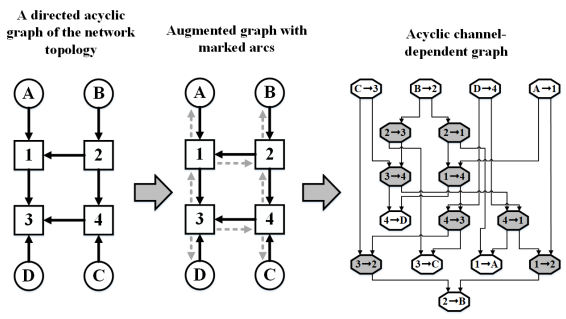

Fig. 1. The process of deadlock-free routes obtaining

To remove deadlocks from the network, circular resource dependencies should be removed. A virtual channel represents a resource. When a packet is transmitted over the network, it occupies a virtual channel for each hop. As the packet passes through the network, the tail of the packet releases the previously occupied virtual channels. The situation when the packet header is blocked because the next virtual channel is transmitting the same packet is called a deadlock. As a result, for wormhole routing the packet header can meet his "body" and a deadlock occurs. A similar situation is possible when there are many packets block each other in a router [13].

There are three main strategies for dealing with deadlocks:

- Strategy for deadlock prevention. Assumes the use of resources in such a way that a deadlock never occurs [14].
- Strategy for deadlock avoidance. Avoiding of deadlocks during the tracking of a route. Resources are provided to the packet only as it is transmitted through the network and if this does not lead to deadlocks. This class includes the Up / Down Routing algorithm, which allows creating routing tables for each switch [14].
- Recovery strategy. Restoring packet transmission in case of deadlocks. Resources are provided to packets without preliminary checking. If deadlocks are found, the resources assigned to other packets [14].

Due to the fact that the Up/Down Routing algorithm is implemented in the second SANDS component, there is no need to take into account the priorities and intensity of information flows. The algorithm avoids loops without blocking of switches and channels for specific information flows. However, the algorithm creates a graph in which cycles are prevented in advance. No matter the packets intensity, packet type or its route, it would not cause a deadlock for this packet.

D. *Transmission delays calculation in SANDS*

In addition to tracking of routes, the SANDS second component provides the user with an estimation of the packets transmission time for each information stream. This is also a very important and complex task. Data transfer delays depend on not only the physical parameters of the network, but also on how much the data transfer will interfere with other flows passing through the common parts of the route. The latency in the second component is calculated by obtaining the worst buffer states on the routed switches. Each switch calculates the time it takes for a packet to pass through the worst-case buffer state. The calculated delay is then summed over the path of the entire route to obtain the resulting delay. This method allows to get the maximum latency when there can be multiple packets in the same packet stream on the network. The issue of calculating delays was discussed in detail in the paper [5]. This research was continued in a separate paper [15], where the analytical model for calculating delays has received a new refinement and the provided calculation example illustrates the calculation.

III. METHOD FOR CHOOSING OF A DEADLOCK-FREE ROUTES CONFIGURATION

In the current implementation of the SANDS, within the framework of the second component, the user gets a set of routes for each information stream by clicking on the "Calculate routes" button. The first available route is shown for each flow for the user. And the transmission delays are calculated for this current configuration.

It is necessary to automate the process of obtaining the most efficient route configuration, develop an algorithm, implement and test it in SANDS software.

In a previous paper [7], it was determined that the main criterion for choosing of a configuration of deadlock-free routes is the total time on all main configuration routes in the network. An auxiliary selection criterion is the transmission time of each route. These criteria should be minimized. It should also be taken into account that network simulation is the task of the other software component of SANDS, so when tracking of routes a tool does not have the simulation results. In the SANDS second component it is assumed that the data streams along the routes go simultaneously and once. Moreover, the second component does not take into account the data priority. All data streams of the network are considered is common data streams with the same priority. A flow in this case is information that is transmitted from node to node, the data that is not generated by the nodes is not considered.

The new method should:

- Choose the best configuration in a reasonable time;
- Have adjustable solution accuracy;
- Be abstract and not tied to a specific delay calculation algorithm;
- Be simple;
- Be compatible with the Up/Down Routing algorithm of the deadlock-free routing;
- Work with the fixed network structure of the SpaceWire standard.

A. *Finding the solution*

Previously, we considered such methods for solving this problem as:

- Method for evaluating routes by logical-probabilistic method;
- Complex method of route selection;
- BGP Best Path Algorithm;
- Method based on routes derived from genetic algorithm;
- Method of reliefs.

Interim results of this study are published in [16].

The route assessment method using a logical-probabilistic methodology is based primarily on the assessment of the frequently changing traffic situation with subscribers - cars and transmitters - radio towers.

An integrated routing method evaluates routes based on noise and transmission quality data.

Border Gateway Protocol (BGP) is a routing protocol between standalone systems. A standalone system is a group of networks or a network with common routing policies and common administration. BGP is mainly used to exchange routing information for the Internet.

Method based on routes derived from genetic algorithm [17] allows you to synthesize sets of routes and compare them with deadlock-free routes built earlier by the Up/Down Routing algorithm.

The relief method using the mathematical model on graphs allows you to build the shortest routes between the nodes of the graph.

Each of them was assessed according to five criteria. The comparison result is presented in the paper [15]. None of the methods were suitable for solving the problem.

Using the considered methods and Dijkstra's algorithm, five methods were formed to solve the problem. Dijkstra's algorithm is a graph-based algorithm [18]. It finds the shortest distance from one of the vertices of the graph to all the others; it works only for graphs without edges with negative weight. Dijkstra's algorithm belongs to the so-called "greedy" algorithms. These are algorithms in which locally optimal decisions are made at each stage, assuming that the final solution also turns out to be optimal [19].

The algorithm consists of 7 steps:

1) All vertices, except the first one, are assigned a weight equal to infinity, and the first vertex - 0.
2) All vertices are not marked.
3) The first peak is declared as current.
4) The weight of all unmarked vertices is recalculated according to the formula: the weight of an unmarked vertex is the minimum number of the old weight of the given vertex, the sum of the weight of the current vertex, and the weight of the edge connecting the current vertex with the unmarked one.
5) Among the unmarked vertices, the vertex with the minimum weight is found. If one is not found (the weight of all vertices is infinity), then the route does not exist – exit the algorithm. Otherwise, the found vertex becomes the current one and marked.
6) If the current vertex is the final one, then the path is found, and its weight is the weight of the final vertex.
7) Return to step 4.

The algorithm operation ends when it reaches the final vertex, and the weight of the shortest path becomes the weight of the final vertex [20].

Advantages of the algorithm:

- Simple and proven algorithm.

Disadvantages of the algorithm:

- Does not take into account delays;
- Has no adjustable solution accuracy;
- Does not take into account deadlocks.

Other similar tasks, which have their own methods of solution, were also considered. One of these tasks was the "Backpack Problem". This is an NP-complete problem. Many items are given, each has a certain weight and value. It is necessary to choose such a set of items so that the total weight does not exceed the capacity of the backpack. The total cost of the selected items should be maximum [21].

The problem has exact methods of solution - brute force and branch-and-bound method. There are also approximate solution methods. Examples include various genetic algorithms, greedy algorithms, and others. However, such a task is completely different from the current task of the paper. Indeed, it is necessary to find a combination. However, making an analogy, the items in the backpack will change their weight each time a new combination is selected. Therefore, this task and methods for its solution will not work.

Another problem considered is the Steiner problem [22]. The goal is to find the shortest network connecting a given finite set of points. There is no effective algorithms that gives an exact solution of a problem. An approximate solution is given by Kruskal's algorithm. However, such an algorithm is created for a weighted connected undirected graph, which is not suitable for current task since the CDG graph of the constructed network is necessarily directed. Significant changes would make the tracked routes not deadlock-free. In addition, routes will be discarded in a suboptimal way. In addition to the presented ones, other problems and methods of their solution were considered. They also did not fit for various reasons. The results formed the basis of a new study in the paper [7]. As a result, five methods were presented for solving the problem of choosing the best configuration of deadlock-free routes:

- The first method, based on a brute force of all possible options (Method I);
- The second method is based on the BGP algorithm, which allows to analyze the specified criteria and select the configuration of the routes step by step (Method II);
- The third method based on game theory and Nesh-Wardrop equilibrium (Method III);
- The fourth method, based on labeling the CDG graph by the number of flows passing through each vertex of the graph and using Dijkstra's algorithm (Method IV);
- The fifth method allows an iterative approach to solving the problem. Sets the labeling of the CDG graph by the number of passing routes through each vertex of the graph (Method V).

Let us compare the considered methods, analyze their suitability for choosing of a configuration of deadlock-free routes, assess the advantages and disadvantages. Each method should be evaluated according to the 5 parameters:

1) Reasonable execution time for different network configurations;
2) Adjustable solution accuracy;
3) Abstractness from a specific method for calculating of latencies;
4) Simplicity;
5) The main selection criterion is the total transmission time for all main configuration routes in the network and the transmission time for each route;

According to Table 2 Method V is best to solve the problem. However, we should evaluate the advantages and disadvantages that are not listed in the table for each of the proposed methods. This will help to provide analysis that is more detailed and confident.

TABLE II. THE RESULT OF COMPARING OF THE PROPOSED METHODS

Parameters:	1	2	3	4	5
Method I	-	-	-	+	+
Method II	+	-	-	+	-
Method III	+	-	+	-	+
Method IV	+	-	+	+	+
Method V	+	+	+	+	+

As a result of the analysis, the iterative approach (Method V) was chosen as the best method for choosing the configuration of SpaceWire deadlock-free routes. This method and its modifications was presented in the paper [7].

B. Theoretical justification of the chosen method

Within the framework of the functionality of the SANDS tool, the following assumptions are made:

1) Information flows have no priority;
2) Flows between network nodes are considered only;
3) Transfer of data streams start at the same time;
4) Data transfer takes place only once.

According to the first assumption, only large information flows are taken into account. The service information on the network is rather small and its presence has practically no effect on data transfer. The tasks of the SANDS second component include routing of large information flows between nodes in the network. The second assumption helps analyze the worst-case network data transfer. If all information flows simultaneously start transmitting information packets, the network will have the maximum load. Thus, the worst case for data transmission is considered. There is no need to consider retransmission of packets, since latency is calculated by a separate mechanism, and a separate SANDS component is involved in simulation of the network operation.

The method can be applied to any network topology. Due to the mechanism of distribution of weights on graphs, it does not matter how nodes and switches are connected. Let the network topology be represented as a graph G = (V, E), where V is the set of nodes and switches in the network, E is the set of communication channels. Then G ’(V, A) denotes the directed graph of the network topology. Moreover, V are network nodes, and A are oriented arcs between nodes.

Channel-dependent graph (CDG graph) displays the dependence of the channels occupied by packets during data transmission in the network. A channel-dependent graph contains vertices that represent the physical channels of the network, and arcs that determine the possible sequence of using these channels. CDG graph will be denoted as: G ’’ (V ’, A’), where V ’ is the set of nodes of the channel-dependent graph, A’ is the set of arcs of the channel-dependent graph. CDG graph contains all channels of graph G ’, and the vertices of graph G ’’ are oriented arcs A of graph G ’.

According to Equation 1, for each vertex of the CDG graph, the weight is calculated by the number of routes passing through it.

$$w(i) = \sum_{j=0}^{N} M_j \quad (1)$$

Where, w() is the function for calculating of the weight of the i-th vertex from the list, N is the number of vertices of the CDG graph in the route, M is the route passing through the j-th vertex of the CDG graph.

A similar calculation is performed for each vertex V ’ from graph G ’’. However, each vertex of such a graph corresponds to the directed arcs A of the network topology graph. SpaceWire standard supports the bi-directional channels. Thus, the set of directed arcs A is only the directions in which the channels of the topology graph of the network G are used. Thus, by making a marking of the CDG graph we get a marking of the network topology graph. Any network structure can be represented as a graph. Thus, the method is applicable to any topology.

The method of choosing the configuration of deadlock-free routes does not guarantee long-term convergence. At first, the method will converge as the phantoms are removed from the network and the nodes weights decrease. When all phantoms have been removed, the search for the best configuration will begin. During the search, the total value of the weights of all the main routes throughout the graph will change as the iterations go through. Its values will fluctuate around a certain range, replacing each other. Having reached the iteration threshold specified by the user, the iteration should stop. The answer is the route configuration obtained at the iteration with the lowest total weight of all the main routes in the graph. This will allow receiving a response with the lowest overall network latency.

Search does not guarantee an optimal solution. However, most of the worst-case scenarios will be discarded by using the mechanism for calculating the weights in the graph and obtaining a weighted score for each route. The algorithm simply does not consider unprofitable routes.

Let us make k iterations according to the value specified by the user. As an answer algorithm chooses the distribution obtained at the iteration with the smallest value of all the main routes total weight. This will be the best choice with the lowest possible overall network latency. The expected result is shown in Fig. 2.

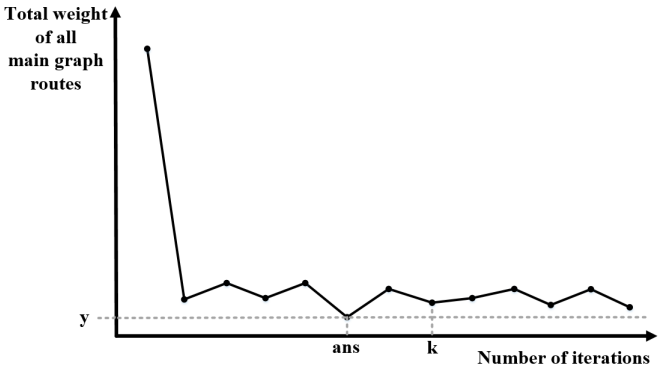

Fig. 2. Expected graph of the method operation results

The more streams are transmitted along one route, the longer the transmission will take, since the channel will be occupied first by one packet, then by the second one following it, and so on. In contrast, the fewer packets transmitted over the channel, the faster the transmission will finish. This situation is taken into account during the graph marking. The less weight on the CDG graph node, the faster it will be possible to transfer the data. The method uniformly distributes the weights on the graph corresponding to the routes. It makes possible the uniform distribution of the network load during data transmission. This method does not overload parts of the network and avoids long waiting. Only an approximate optimal solution is provided within the considered iterations. Most of the inefficient routes will be discarded. Indeed, at the first iteration, the method weighs each route for the information flow, the routes with the highest weight are discarded. These are primarily the longest or most congested routes. Further enumeration changes the decision only slightly, since the longest routes will still carry more weight relative to their more efficient neighbors.

IV. Verification of the Developed Method

A. Test scenarios for experiments

The previously selected technique was implemented in the SANDS component of deadlock-free routing. Test scripts are needed to test the functionality of the implemented technique. The technique should be tested on three circuits of on-board networks.

The first example of a network consists of three nodes and six switches. It is a simple data transmission network with a ring topology. The network diagram implemented in the SANDS software is shown in Fig. 3.

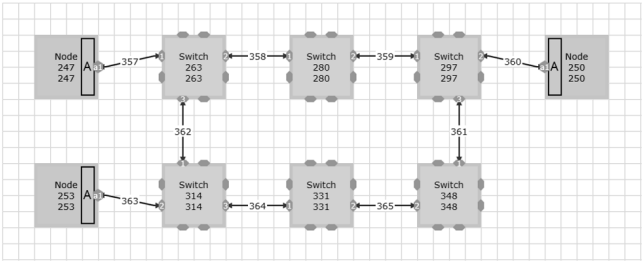

Fig. 3. The first diagram of a data transmission network with a "ring" topology in the SANDS

In a given network, six data packets are transmitted. These are ordinary messages with a Boolean addressing type.

Three packets of data are sent from node 247 to node 253. Three more packets from node 250 to node 253. The second diagram of the data transmission network is shown in Fig. 4.

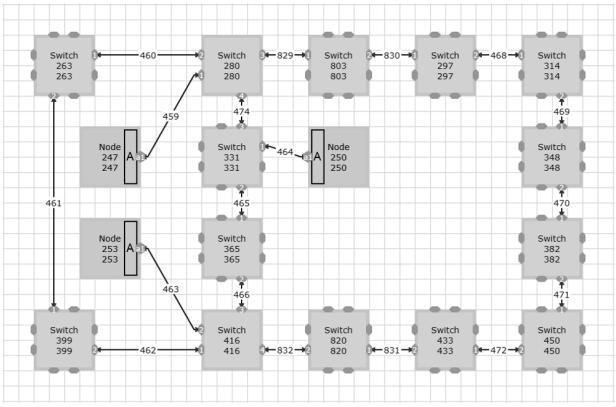

Fig. 4. The second diagram of a data transmission network with a complex topology in the SANDS

The network consists of 3 nodes and 14 switches. In a given network, six data packets are transmitted. These are ordinary messages with a Boolean addressing type. The principle of packet transmission is the same as the previous example.

In total, three packets are transmitted from 247 and 250 nodes to node 253, respectively. The third network diagram has a 4x6-grid topology. An example of a network diagram is shown in Fig. 5.

There are 12 nodes and 24 switches in the network. In the presented network, 132 packets are transmitted. The network also transmits ordinary packets with logical data addressing. Each node is assigned its own logical address from 32 to 43. Each node transmits 11 packets to all other nodes.

B. Experimental results

Current chapter presents the testing of a developed and modified methods according to the presented scenarios. The modified method is represented by the modified calculation of the graph weights. In the basic technique, the number of routes measures the weight of each node. In the modified technique, the weight of each node is equal to the number of routes. The basic and modified algorithm was tested separately.

C. *Testing the network scheme with the "ring" topology*

Testing begins with the first network diagram, testing the basic method for selecting route configurations. Figure 6 shoes the loaded network topology in the SANDS software and the interface window of the second component. After clicking the "Calculate routes" button, the Up/Down Routing algorithm is launched. It successfully builds deadlock-free routes on a given ring topology. After that, the implemented configuration selection method is launched. The user receives the response shown in Fig. 6.

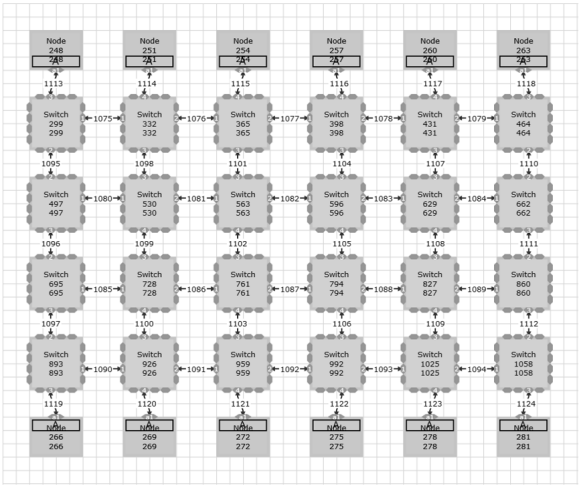

Fig. 5. The third diagram of a data transmission network with a 4x6-grid topology in SANDS

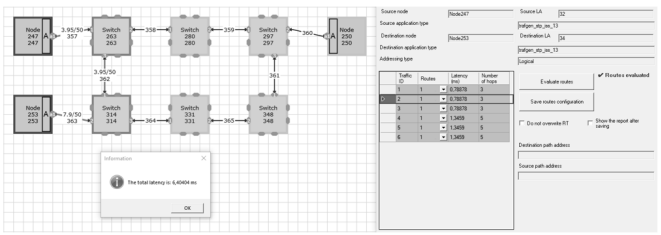

Fig. 6. The result of the basic method on the first scheme with the topology of the "ring" type network

As a result, the total network transmission time is the sum of the transmission delays along each route. It is 6.40404 milliseconds. This value really corresponds to the minimum value. Node 247 transmits the data along the shortest transmission path, which allows data to be transmitted with minimal delay. Node 250 transmits the data along the only available path, affecting only one common data channel with node 247-363. Thanks to this solution, data is transmitted in the best possible way. Changing of at least one route will cause the increasing of latency. The new combination will not be effective enough compared to the result provided by the implemented method. The latency in this case will be 9.05364 milliseconds, which is more than 6.40404 milliseconds.

As a result of the operation of the algorithm, the total value of all weights of the main routes of the CDG graph was written to a separate file at each iteration. According to the result, it is possible to draw up a graph of the change in the total weight of the main routes of the graph at each iteration. The resulting graph is obtained using data from the log and is shown in Fig. 7.

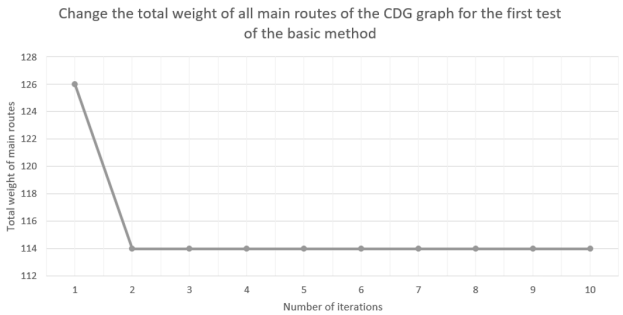

Fig. 7. The resulting graph of weights for the first test of the basic method

The obtained result really corresponds to the expected one. First, there is a sharp decrease in the weight value as a result of the removal of phantom routes, and then algorithm gives the answer. However, this is a simple test case and for more complex topologies the graph may not converge. The method's answer is always the minimum weight received. The operation of the software implementation of the method took 6 milliseconds.

Let us proceed with the results of a modified methodology for choosing a configuration of deadlock-free routes. The result of the operation is shown in Fig. 8.

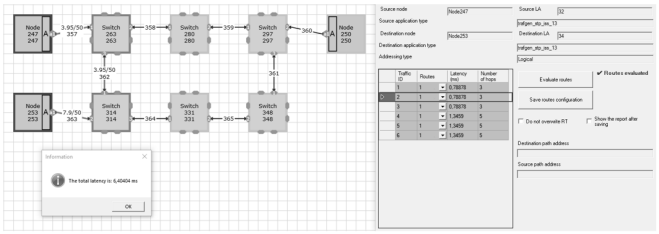

Fig. 8. The result of the modified method on the first network diagram with the "ring" topology

There are no visible differences. The total latency is also 6.40404 milliseconds, the obtained solution is identical. However, the content of the test file has changed. Now the total weight of the main routes in the graph is compiled according to a new principle. Log contains the following values: 114, 114, 114, 114, 114, 114, 114, 114, 114, 114. As a result, it can be seen that the answer was received at the very first iteration of the algorithm. The operation of the software implementation of the method also took 6 milliseconds.

D. *Testing the network scheme with complex network topology*

Testing of the second network diagram is carried out in a similar way. The user receives the response shown in Fig. 9.

As a result of the algorithms for the packets of node 247, three routes are obtained:

1) Short route through channels: 459, 474, 465, 466, 463;
2) Short route through channels: 459, 460, 461, 462, 463;

3) Long route through channels: 459, 829, 830, 468, 460, 470, 471, 472, 831, 832, 463.

However, for packets from node 250, the deadlock-free routing algorithm provides only one transmission path. It goes through the channels: 464, 465, 466, 463.

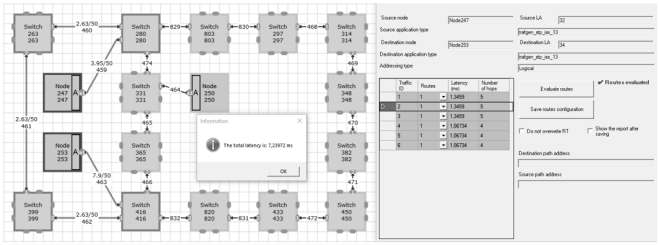

Fig. 9. The result of the basic method on the second scheme with a complex network topology

The routing configuration technique prompts the user to track all routes from node 247 along a second short path. This is indeed the correct decision, since the node 250 has no alternatives and it already uses some of the channels from the first route of the node 247. However, the technique does not suggest the user to use a long and inefficient third data transmission route. This is a correct result.

The total latency for the combination suggested by the SANDS is 7.23972 ms. This delay is minimal for the considered data transmission network, for given data streams. If the user decides to change the route to another, then the delay will increase. As a result of the algorithm operation, the total value of all weights of the main routes of the CDG graph was written to a separate file at each iteration. The resulting graph obtained using data from a log file and presented in Fig. 10.

The expected result of changing the total weight of the main routes of the graph is shown in Figure 5. The result really corresponds to the expected one. At the first iterations, a sharp decrease in the weight value is observed as a result of the removal of phantom routes. Then the data fluctuates around a certain value. As a result, at the fifth iteration, the minimum value is obtained, so we got the best route configuration for a given network. The operation of the basic method took 7 milliseconds.

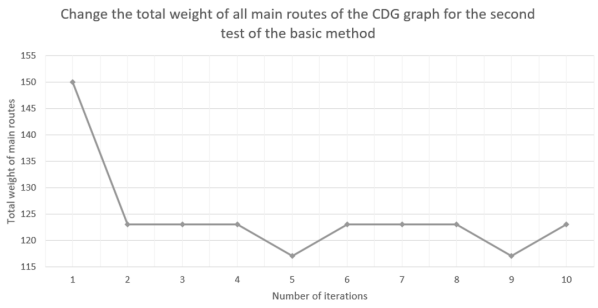

Fig. 10. The resulting graph of weights for the second test of the basic method

Testing of the modified method gave a solution identical to the original. The total latency was 7.23972 milliseconds.

The answer was received in the same way at the fifth iteration, as in the original solution. The graph of the second test for the modified method is shown in Fig. 11.

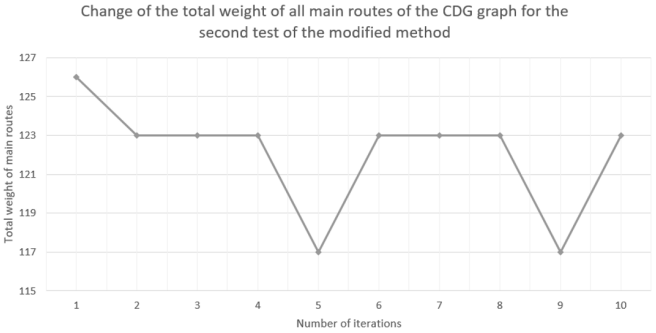

Fig. 11. The resulting graph of weights for the second test of the modified method

According to the measurements, the operation of the modified method took 7 milliseconds.

E. Testing the scheme with the «grid» network topology

The process is similar to the previous examples; user loads the topology and opens the interface window of the SANDS second component. After pressing the button "Calculate routes" deadlock-free routes are built. The basic method of choosing a route configuration gives the answer shown in Fig. 12.

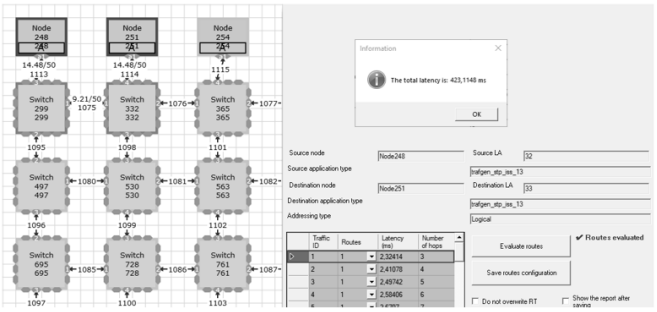

Fig. 12. The result of the basic method on the third scheme of the network with the 4x6-grid topology

Current network has the bigger size and the large number of data streams. As a result, the total latency of all major routes is 423.1148 ms.

The chosen configuration is one of the best route choices. Fig. 12 shows the first eleven flows of node 248 transmitting packets to their destinations in the shortest possible way. If the user tries to manually change the configuration, the result will be worse. The total latency will rise to 431.90584 ms. The total weight of all major routes in the CDG graph has grown significantly with the growth in the number of data streams.

Fig. 13 shows the graph based on the results of the algorithm operation. There is a significant change in the first iteration and the minimum value was obtained at the eighth iteration.

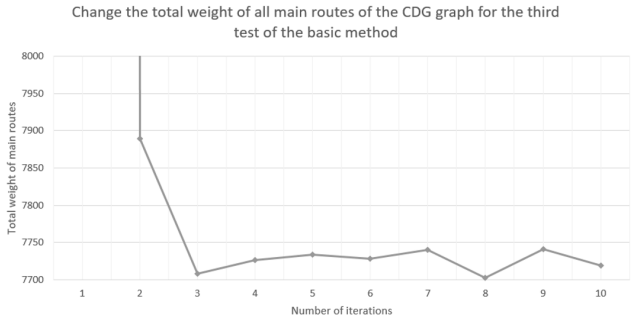

Fig. 13. The resulting graph of weights for the third test of the basic method

The answer is indeed one of the best configurations that can be made from the given deadlock-free routes. The method has simply discarded most of the worst-case scenarios and long routes, leaving the most efficient configuration. As a result of the measurements, the basic method implemented in the SANDS software worked for 70 milliseconds on a large circuit with a 4x6-grid topology.

For the modified version of the algorithm, the total latency is 435.60824 milliseconds. This is more than what was obtained using the basic technique.

The answer was received at the seventh iteration, which is one iteration earlier than with the basic approach. The graph of the third test for the modified method is shown in Fig. 14.

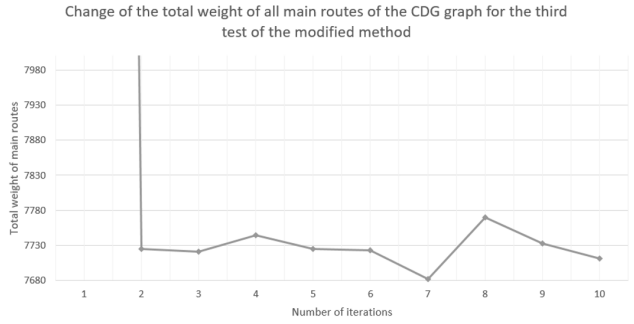

Fig. 14. The resulting graph of weights for the third test of the modified method

The modified version also took 70 milliseconds of operation.

F. *Evaluation of results*

After the implementation and execution of a tests, the results should be summarized and analyzed. First of all, it is worth assessing the dependence of the behavior of the basic technique on networks of different sizes. The estimation is carried out by comparing the graphs of the main routes total weights. The result for the basic method is shown in Fig. 15.

The rate at which a response is received by the basic method depends on the size of the network. With the same number of data streams, a better combination will be found for a network with fewer switches and a simpler network topology. Indeed, the fewer links and switches in the network, the fewer routes will be offered to flows. In turn, the weights at the vertices of the graph will be less. Streams will be able to quickly determine their choice. To compare the modified methodology on small and medium-sized networks, it is worth building a separate graph. The result of construction is shown in Fig. 16.

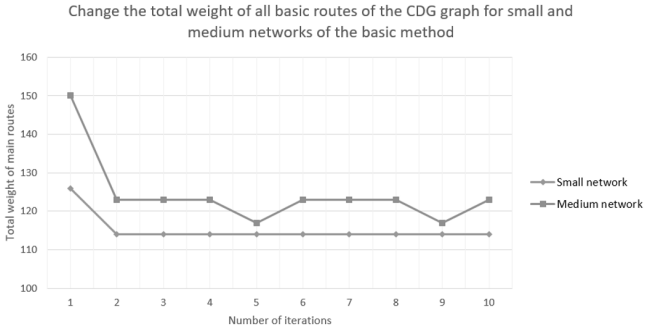

Fig. 15. Comparison of the basic method on small and medium networks

For the modified method, the same dependence is observed as for the original one. The smaller the network, the faster the result is received. However, thanks to the modified method, the values at the vertices of the graph have a smaller scatter compared to the basic method. Thus, the answer can be obtained at the first iterations. However, further behavior is comparable to the basic technique. The total weight continues to fluctuate around a certain value. This is clearly seen when comparing Fig. 15 and 16. Thus, the modified technique allows to achieve the correct answer quickly if you need to analyze simple transmission networks.

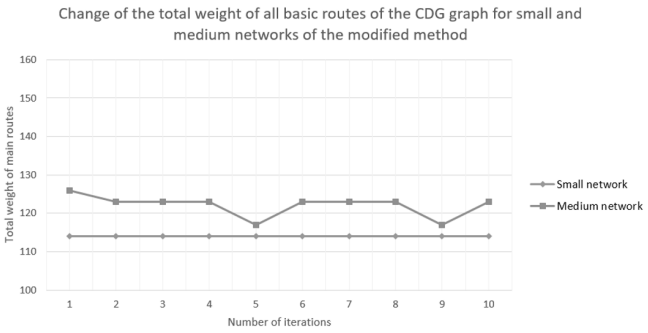

Fig. 16. Comparison of the modified method on small and medium networks

Comparison of the basic and modified method on large networks is shown in Fig. 17.

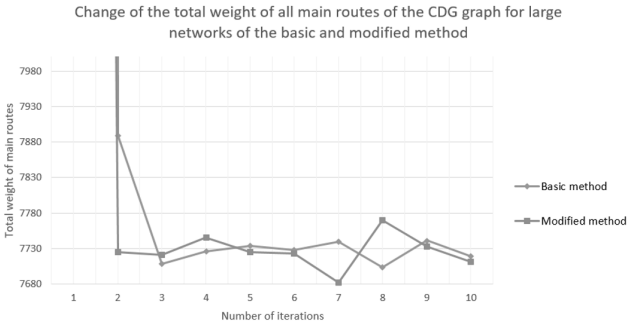

Fig. 17. Comparison of the basic and modified method on large networks

The modified method has a larger range in which the search for the route configuration is performed. The solution of the modified method is performed in the range from 7680 to 7780. In contrast to the basic method, in which the search for values has a narrower range. The best configuration was obtained using the basic method. The basic method gradually moves to the answer, reducing the values of the total weight of the main routes, in contrast to the modified one.

For a more detailed analysis, it is worth evaluating the result of the work of the second component without using the implemented methodology. Initially the user is offered a configuration consisting of open-ended routes that were built first according to the "Up/Down Routing" algorithm.

Table IV shows the results of measurements of the total latency of all major configuration routes for different types of networks and methods.

TABLE III. COMPARISON OF THE RESULTS OF THE SECOND COMPONENT

Network type:	Small network with «ring» topology	Medium network with complex topology	Large «4x6 grid» topology
No method	6,40404 ms	7,23972 ms	583,21336 ms
Basic method	6,40404 ms	7,23972 ms	423,1148 ms
Modified method	6,40404 ms	7,23972 ms	435,60824 ms

According to the table, there is no significant difference when working with simple networks and a small number of threads. However, there are significant benefit for large networks. Although the modified method gets a slightly worse response than the basic one, other network topologies are possible where the modified method may get better results. Applying the implemented techniques really allows to get a better answer than what was originally suggested by the SANDS.

There was no difference for time spent for operation of basic and modified versions. The computation time for each iteration is the same when using different methods. The implemented methods showed the user have to wait for the result:

- 6 milliseconds for small networks with a simple topologies and a small number of data flows;
- 7 milliseconds for medium networks with complex network topology and few data flows;
- 70 milliseconds for large networks with complex network topology and large number of data flows.

In general, the time that user needs to get the result is acceptable. This result is certainly suitable for implementation in SANDS software.

The developed method and its modification are:

- Choose the best configuration in a reasonable time;
- Have adjustable solution accuracy. Thanks to the iterative approach, the user can adjust the number of iterations affecting the waiting time and the accuracy of the result;
- Quite abstract and not tied to a specific algorithm for latencies calculation. This will allow the method to operate correctly even when the principle of the calculation of delays changes, according to the requirements of the customers;
- Quite simple and quick to implement;
- Compatible with the Up/Down Routing algorithm;
- Works with the fixed network structure of the SpaceWire standard;
- The main selection criterion is the total transmission time on all main configuration routes in the network. An auxiliary selection criterion is the transmission time of each route. Techniques allow routs to choose the best solutions for themselves. These criteria are really minimized within the framework of the iterations.

V. CONCLUSION

The provided research result is the new method for the choosing of the deadlock-free routes in the onboard networks. To obtain the result, we overviewed the main methods and algorithms for tracking of deadlock-free routes and chose the base algorithm of deadlock-free routes analysis - Up / Down Routing algorithm. The results of the algorithm is a set of deadlock-free routes for the network configuration, but still there is a problem, how to get the best routes configuration. So we provided an overview and evaluation of existing methods for choosing of the best configuration. The evaluation of methods showed that a new method should be developed. WAs a result, five different methods for solving the problem were formulated and the compared. Assessment of the developed methods made possible to choose an iterative method with an estimation of the CDG graph as the best one.

The chosen method and its modification were implemented in C # in the SANDS software. Paper provides a few experimental topologies of on-board networks to verify the efficiency of the implemented method. We verified both the basic method and its modification on each of the presented topologies.

Testing has shown that both methods solve the main task of the research. They do find a better route configuration compared to the basic functionality of the deadlock-free route component. In addition, each of the implemented methods search for a solution very quickly and it is proved for a variety of network structure types. The algorithm operation time measured during the experiments fully meets the requirements.

As a result, both methods have shown excellent results. Each of them is suitable for different types of network structures. The basic method and its modification are implemented in the SANDS software.

ACKNOWLEDGMENT

This work was prepared with the financial support of the Ministry of Science and Higher Education of the Russian Federation, Grant Agreement No. FSRF-2020-0004 “Scientific Foundations for the Development of Architectures and

Communication Systems of New Generation On-Board Information and Computer Systems in aviation, space systems and unmanned vehicles".

REFERENCES

[1] S. Balandin, S. Virtanen, M. Gillet, I. Lavrovskaya, V. Olenev, A. Rabin, A. Stepanov, *Co-Modeling of Embedded Networks Using SystemC and SDL. International Journal of Embedded and Real-Time Communication Systems (IJERTCS).* 2013. 23-48.

[2] V. Olenev, I. Lavrovskaya, L. Kurbanov, I. Korobkov, Y. Sheinin. *Computer-aided design of onboard space networks. Radio electronics issues.* No 8, 2018. 145-153.

[3] *ESA. Standard ECSS-E-ST-50-52C, SpaceWire — Remote memory access protocol. Noordwijk: Publications Division ESTEC,* February 5, 2010.

[4] Y. Sheynin, V. Olenev, I. Lavrovskaya, I. Korobkov, S. Kochura, V. Shkolniy, D. Dymov. *Computer-Aided Design System for On-board SpaceWire Networks Simulation and Design, Proceeding of the 20th conference of FRUCT association,* 2017. 1-6.

[5] V. Olenev, A. Karandashev, K. Alexeeva. *Tools for analysis and tracking of deadlock-free routes in on-board SpaceWire networks, Finnish-Russian University Cooperation in Telecommunications (FRUCT 26th),* 2020. Pp. 628-634.

[6] V. Olenev, A. Karandashev. *Methods for selecting the optimal configuration of deadlock-free routes in on-board SpaceWire networks, Aerospace Instrumentation and Operational Technologies, Second International Scientific Conference,* 2021. Pp.293-302.

[7] A. Karandashev, V. Olenev. *Selection Methods for Deadlock-free Routes' Optimal Configuration in On-board SpaceWire Networks, Wave Electronics and its Application in Information and Telecommunication Systems (WECONF).* 2021.

[8] Y. Sheynin, V. Olenev, I. Lavrovskaya, I. Korobkov, D. Dymov. *STP-ISS Transport Protocol for Spacecraft On-board Networks, Proceedings of 6th International SpaceWire Conference* 2014 Program; Greece, Athens, 2014. Pp. 45-76.

[9] V. Olenev, Y. Sheynin, I. Korobkov, E. Suvorova, E. Podgornova, D. Dymov, S. Kochura, I. Lavrovskaya*, STP-ISS transport protocol for SpaceWire on-board networks: Development and evolution. 5. International Journal of Embedded and Real-Time Communication Systems (IJERTCS).* 2014. Pp. 45-76.

[10] V. Olenev, Y. Sheinin, N. Sinyov, I. Korobkov, I. Lavrovskaya. *Hierarchical Simulation of Onboard Networks, Intelligent Distributed Computing XIII. IDC 2019. Studies in Computational Intelligence,* 2019. Pp. 191-196.

[11] J. Dally, L. Seitz*. Deadlock-free message routing in multiprocessor interconnection networks,* 1988. Pp. 1-15.

[12] C. Sancho, A. Robles, J. Duato. *A new methodology to compute deadlock-free routing tables for irregular networks, International Workshop on Communication, Architecture, and Applications for Network-Based Parallel Computing,* 2000. Pp. 45-60.

[13] S. Warnakulasuriya, T. Pinkston. *Characterization of deadlocks in interconnection networks, Parallel Processing Symposium,* 11th International. – IEEE. 1997. Pp. 904-921.

[14] J. Duato, S. Yalamanchili, L. Ni. *Interconnection networks: an engineering approach.– Morgan Kaufmann,* 2003. Pp. 1025.

[15] K. Alexeeva, V. Olenev. *Analytical model for calculating data packet transmission delays in SpaceWire onboard networks, Wave Electronics and its Application in Information and Telecommunication Systems (WECONF),* 2021. Pp. 261-264.

[16] Olenev, V., Karandashev, A. *Configuring Methods for Deadlock-Free Routing. International Journal of Embedded and Real-Time Communication Systems (IJERTCS),* 13(1), 2022. c. 1-20.

[17] D. Zaichenko, I. Sineva, Irina. *The Study of Genetic Type Steganographic Models to Increase Noise Immunity of IoT Systems. International Journal of Embedded and Real-Time Communication Systems. 11. International Journal of Embedded and Real-Time Communication Systems (IJERTCS).* 2020. Pp. 1-15.

[18] W. Dijkstra*. A note on two problems in connexion with graphs, Numerische Mathematik,* 1959. Pp. 269-271.

[19] S. Lebedev, F. Novikov. *Necessary and sufficient condition applicability of the Dykstra algorithm, Computer tools in education,* 2017. Pp. 5-13.

[20] G. Vanykina, T. Sundukova*. Structures and algorithms for computer data processing,* INTUIT. 2011.

[21] I. Sigal. *Knapsack problem: Theory and computational algorithms, Textbook for the course "Discrete Mathematics".* 1999.

[22] I. Romanovsky*. Steiner's problem on graphs and dynamic programming, Computer tools in education,* 2004. Pp. 80-86.