УДК 651.3:518.5

П.І. ФЕДОРУК

ГРАФО-АВТОМАТНА МОДЕЛЬ АДАПТИВНОЇ СИСТЕМИ ДИСТАНЦІЙНОГО НАВЧАННЯ ТА КОНТРОЛЮ ЗНАНЬ

Abstract: *The article reviews the usage of graph-automated model of adaptive system of distance learning and control of knowledge for solving the task of optimal control of quantum stream of knowledge in the process of receiving the course by students. The model under study takes into account such important aspects of distance learning as possibility of "forgetting" the knowledge quantum by a student: the problem of the order of presenting of quantum for working and operating on: repetition of certain quantum in different classes and contexts. Described model of quantum stream control lets to realize in the system of distance learning and knowledge control of certain possibilities of adaptation to peculiarities of perception and knowledge levels of those who are studying.*
Key words: *system of distance learning, adaptive process, graph-automated model, quantum of knowledge.*

Анотація: *У статті описано застосування графо-автоматної моделі адаптивної системи дистанційного навчання та контролю знань для вирішення задачі оптимального управління потоком квантів знань у процесі опрацювання курсу студентом. Розглянута модель враховує такі важливі аспекти дистанційного навчання, як можливість «забування» квантів знань студентом, проблему черговості виведення квантів на опрацювання та виконання над ними операцій, повторюваність одних і тих же квантів у різних заняттях і контекстах. Описана модель управління потоком квантів дозволяє реалізувати в системі дистанційного навчання і контролю знань значні можливості адаптації до особливостей сприйняття і рівнів знань тих, хто навчається.*
Ключові слова: *система дистанційного навчання, адаптивний процес, графо-автоматна модель, квант знань.*

Аннотация: *В статье описано применение графо-автоматной модели адаптивной системы дистанционного образования и контроля знаний для решения задачи оптимального управления потоком квантов знаний в процессе усвоения курса студентом. Рассмотренная модель учитывает такие важные аспекты дистанционного образования, как возможность "забывания" квантов знаний студентом, проблему очерёдности выведения квантов на обработку и исполнения над ними операций, повторяемость одних и тех же квантов в разных занятиях и контекстах. Описанная модель управления потоком квантов позволяет реализовывать в системе дистанционного образования и контроля знаний значительные возможности адаптации к особенностям восприятия и уровням знаний обучающихся.*
Ключевые слова: *система дистанционного образования, адаптивный процесс, графо-автоматная модель, квант знаний.*

1. Вступ

Поєднання інформаційних технологій та інноваційних педагогічних методик здатне підвищити ефективність і якість освітніх програм, підсилити адаптивність системи освіти до особливостей сприйняття і рівнів знань тих, хто навчається. На сучасному етапі розвитку освіти для цього в основному використовуються адаптивні системи навчання, які базуються на інформаційних технологіях. Це сприяє створенню найбільш сприятливого середовища для розвитку студентів з виявленою обдарованістю і міцною основою для побудови дидактичної системи розвитку потенціалу, а також дозволяє враховувати вікові й індивідуальні особливості. Використання сучасних інформаційних технологій у навчальному процесі дозволяє підвищити якість навчального процесу і підсилити освітні ефекти від застосування інноваційних педагогічних програм і методик, оскільки дає викладачам додаткові можливості для побудови індивідуальних освітніх траєкторій студентів, а також дозволяє автоматизувати цей процес. Застосування інформаційних технологій дозволяє реалізувати диференційований підхід до студентів з різним рівнем готовності до навчання. Інтерактивні навчальні програми, які базуються на гіпертекстовій структурі та мультимедіа, дають можливість організувати одночасне навчання студентів, які володіють різними здібностями і можливостями. Початок адаптивного навчання – час виникнення педагогічних праць Я.А.

Коменського, І.Г. Песталоцци і А.В. Дістервега. А.В. Дістервег: “Викладай відповідно до природи... Вчи без прогалин... Починай викладання з того, на чому зупинився учень... Без знання того, на чому зупинився учень, неможливо добре навчити його” [1]. Сама ідея, яка існувала вже давно, в сьогоднішній інтерпретації з використанням сучасного рівня інформаційних технологій може набути практичної реалізації і автоматизації. Тому створення моделі для побудови адаптивної системи навчання є важливою і актуальною задачею.

2. Адаптивна система дистанційного навчання

Традиційно навчальний курс розділяють на логічні частини – заняття. Проте для реалізації адаптивних можливостей в системі дистанційного навчання заняття пропонується розділяти на менші частини – кванти знань. Квант – найменша неподільна смислова порція інформації (первісне поняття, ключове слово, аксіома, означення, тощо) [2, 3]. Кожен квант може використовуватися в різних заняттях.

У процесі навчання над квантами виконуються операції. Одна і та ж операція може виконуватись над різними квантами. Перед виконанням операції над квантом він мусить бути опрацьованим, тобто вивченим, засвоєним студентом. Таким чином, навчальний курс може бути представлений як множина квантів, які треба засвоїти, і множина операцій, які треба виконати над квантами для їх засвоєння. Кожному кванту відповідає декілька операцій і, навпаки, одну і ту ж операцію можна виконувати над різними квантами.

Отже, адаптивність процесу навчання буде представлена в послідовності подання квантів на опрацювання, чергуванні виконання операцій над квантами та їх опрацюванні, присутності кванта в певному занятті.

Слід зазначити, що можливі два способи прив’язки певного кванта до певного заняття. В першому випадку заняття визначається набором квантів, які мають бути засвоєні. На основі присутності квантів в занятті визначається набір операцій, які мають бути виконані в цьому занятті. В другому випадку заняття задається набором операцій, а кванти, які мають бути засвоєними, вибираються за зв’язками з відповідними операціями.

При реалізації такого способу адаптивності потрібно створити модель керування та опрацювання потоком квантів. Для цього можна використати різні технології. Наприклад, ієрархічну модель дедуктивних запитів [4], мережі Петрі [5], потокові схеми Келлера [6]. Оцінивши, з точки зору ефективності застосування, дані технології, для реалізації поставленої задачі запропоновано формальну модель з використанням елементів теорії графів [7] та теорії автоматів [8].

3. Графо-автоматна модель адаптивної системи дистанційного навчання

Формально схему управління та опрацювання потоком квантів можна зобразити у вигляді орієнтованого графа QSM (Quantum Stream Management), вершини та ребра якого являють собою структурні автомати різних конструкцій (в залежності від призначення вершини чи ребра).

Структурним автоматом вважаємо мережу автоматів, вхід якої має n каналів $(\mathrm{X}_1...\mathrm{X}_\mathrm{n})$, вихід – p каналів $(Y_1...Y_p)$ [7]. Внутрішня пам’ять структурного автомата складається з k елементів

$(Q_1...Q_k)$. Кожному елементу пам'яті відповідає функція переходу $\varphi_1...\varphi_k$, а кожному вихідному каналу – функція виходу $(\psi_1..\psi_p)$. В роботі графа використовується поняття «дискретного часу» τ. В кожен момент часу τ на вхідні канали $X_i[\tau]$ структурного автомата поступають числові сигнали, які перетворюються функціями переходу φ_j, і результат запам'ятовується у відповідному елементі пам'яті $Q_j[\tau] = \varphi_j(X_1[\tau],...,X_n[\tau],Q_1[\tau-1],...,Q_k[\tau-1]),\ j=1..k$. Після переобчислень всіх елементів пам'яті за допомогою функцій виходів обчислюються числові значення, які будуть встановлені на вихідних каналах структурного автомата в даний момент дискретного часу τ: $Y_i[\tau] = \psi_i[\tau](X_1[\tau],...,X_n[\tau],Q_1[\tau],...,Q_k[\tau]), i=1..p$.

До структурного автомата будемо застосовувати два види операцій: запис вектора сигналу на вхід автомата та зчитування значення з конкретного виходу автомата.

Нехай AVT – структурний автомат. Визначимо функції:

$G_{AVT}(Y_j)$ – повертає значення з j-го виходу автомата AVT в даний момент дискретного часу τ;

$S_{AVT}(x_1...x_n)$ – подає на входи $(X_1...X_n)$ автомата AVT відповідні значення $(x_1...x_n)$.

Для цієї функції будемо застосовувати скорочений запис $S_{AVT}(X_i = x)$ (подає на вхід X_i автомата AVT значення x, а на інші входи не впливає).

При розгляді елементів графа QSM як автоматів виникає необхідність опрацювати окремий вхідний чи вихідний канал автомата. Для таких ситуацій введемо позначення $[N].K$, де N – структурний автомат, K – вхідний чи вихідний канал.

Граф QSM складається із структурних автоматів, які взаємодіють за допомогою вхідних та вихідних каналів. Передача числового сигналу від автомата A до автомата B відбувається за два моменти дискретного часу. В перший момент на вихідному каналі $[A].Y$ встановлюється значення в результаті роботи відповідної вихідної функції. В другий момент часу це значення переноситься на канал $[B].X$. Зв'язок виходу $[A].Y$ і входу $[B].X$ будемо позначати $[A].Y \to [B].X$.

Розглянемо структуру графа QSM.

QSM =<Nodes,Edges,IO>,

де Nodes – множина вершин графа QSM; Edges – множина ребер графа QSM; IO – множина портів зовнішнього обміну даними із зовнішнім середовищем.

Опишемо множину Edges. $\mathrm{Edges} = Ed_c \bigcup Ed_d$,

де Ed_c – множина орієнтованих управляючих ребер, Ed_d – множина орієнтованих ребер даних.

Орієнтоване ребро направлене від вихідної вершини до вхідної. Управляючі ребра з множини Ed_c можуть перебувати у двох станах: активному і неактивному. Ребро переходить в активний стан

після відповідної команди від вихідної вершини. В неактивний стан ребро переводиться командою вхідної вершини. Введемо поняття часу t знаходження ребра в активному стані як ще одну характеристику управляючого ребра. Для активного ребра $ed \in Ed_c$ час t дорівнює кількості інтервалів дискретного часу τ з моменту активації ed. Для неактивного ребра ця характеристика дорівнює 0.

Як структурний автомат елемент Ed_c має таку схему (рис. 1):

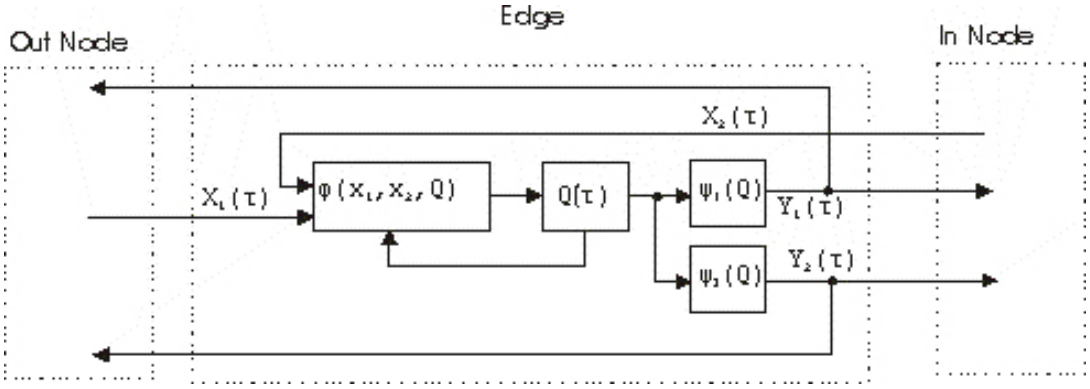


Рис. 1. Схема елемента Ed_c

Тут функції φ, ψ_1, ψ_2 на основі вхідних даних X_1, X_2 та проміжного стану Q в кожен момент дискретного часу τ визначають вихідні сигнали Y_1 та Y_2:

$$Q(\tau) = \varphi(x_1(\tau), x_2(\tau), Q(\tau-1)) =$$
$$= \begin{cases} 0 & x_1(\tau)=0 \vee x_1(\tau)=0 \wedge x_2(\tau)=0 \wedge Q(\tau-1)=0 \\ Q(\tau-1)+1, & x_1(\tau)=1 \wedge x_2(\tau)=1 \vee x_1(\tau)=0 \wedge x_2(\tau)=0 \wedge Q(\tau-1)>0 \end{cases};$$

$$Y_1(\tau) = \psi_1(Q(\tau)) = Q(\tau);$$

$$Y_2(\tau) = \psi_2(Q(\tau)) = \begin{cases} 1, & Q(\tau)>0 \\ 0, & Q(\tau)=0 \end{cases}.$$

З наведених формул видно, що вхід X_1 призначений для активації ребра вихідною вершиною, а вхід X_2 – деактивації вихідною вершиною. Вихід Y_1 своїм значенням має час активності. Вихід Y_2 визначає активність ребра.

Елементи множини Ed_d виконують функцію елементів пам'яті у процесі роботи графа QSM. Характеристикою ребер-елементів Ed_d є вага. Це числова величина, яка встановлюється вихідною вершиною і утримується ребром до наступних змін, незалежно від стану вихідної вершини.

Для опису елементів множини Ed_d як структурних автоматів наведемо таку схему (рис. 2):

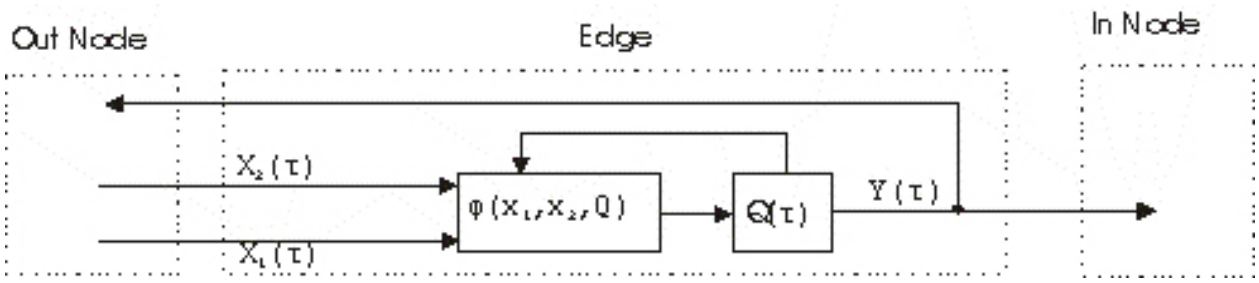


Рис. 2. Схема елементів множини Ed_d

Вихід цього автомата $Y(\tau)$ визначається його станом $Y(\tau) = Q(\tau)$. Стан $Q(\tau)$ в кожен момент часу τ переобчислюється як результат функції φ:

$$Q(\tau) = \varphi(x_1(\tau), x_2(\tau), Q(\tau-1)) = \begin{cases} Q(\tau-1), & x_2(\tau) = 0 \\ x_1(\tau), & x_2(\tau) = 1 \end{cases}.$$

На вхід подаються два сигнали: X_1 – нова вага ребра, ненульове значення каналу X_2 означає необхідність встановити нове значення ваги ребра, тобто X_1. Якщо X_2 дорівнює 0, ваги не міняються.

Для роботи вищеописаного графа потрібні механізми контакту із зовнішнім середовищем Z. Як рішення цієї проблеми використовується IO – множина портів обміну управляючими сигналами та даними із Z. Ці порти використовуються для запуску алгоритму, описаного у графі, отримання сигналів про закінчення, для введення даних та виведення результатів. Порти, які належать множині IO, подібно до ребер (множина Edges), розділені на дві підмножини: управляючі порти та порти даних – є структурними автоматами таких самих конструкцій і мають ті ж характеристики. Елементи IO одним кінцем з'єднані з вершиною, яка належить Nodes, іншим кінцем – із Z. Порти також є орієнтованими, тобто мають початок і кінець. Множина IO складається з чотирьох підмножин: $IO = IO_{d,in} \bigcup IO_{d,out} \bigcup IO_{c,in} \bigcup IO_{c,out}$, де

$IO_{c,in}$ – множина управляючих портів, які мають вхідну вершину у множині Nodes, активуються ззовні;

$IO_{c,out}$ – множина управляючих портів, які мають вихідну вершину у множині Nodes, деактивуються (переходять в неактивний стан) ззовні;

$IO_{d,in}$ – множина портів даних, які мають вхідну вершину у множині Nodes, вага встановлюється ззовні;

$IO_{d,out}$ – множина портів даних, які мають вихідну вершину у множині Nodes.

Основне призначення елементів IO є запуск алгоритму керування потоком квантів та повідомлення про завершення алгоритму. Також елементи IO застосовуються для контакту QSM з механізмом відображення квантів та процесу виконання операцій (інтерфейс студента). Крім того, порти (елементи IO) дають можливість отримувати різноманітну статистичну інформацію про

процес проходження курсу. Наприклад, рівень засвоєння конкретного кванта чи заняття, загальний рівень засвоєння курсу, відсоток опрацьованого матеріалу тощо.

Таким чином, можна сказати, що IO – це механізм обміну інформацією між ядром навчальної системи (QSM) та інтерфейсом студента.

Кожне ребро графа QSM можна описати так:

$$set : sNode \overset{mdf}{\rightarrow} eNode ,$$

де set – множина, до якої належить ребро (Ed_c , Ed_d , $IO_{c,in}$, $IO_{c,out}$, $IO_{d,in}$, $IO_{d,out}$);

$sNode$ – вихідна вершина;

$eNode$ – вхідна вершина;

mdf – модифікатор, використовується у випадках, коли між двома вершинами існує більше, ніж одне ребро одного і того ж типу.

Вершини графа QSM також являють собою структурні автомати. Оскільки вершини за своїм логічним призначенням поділяються на значно більше груп, ніж ребра, то і конструкція відповідних автоматів є різною. Проте для опису вершин графа QSM введемо деякі загальні поняття.

Вершини графа QSM об'єднані у множині Nodes. Вершини характеризуються множинами P_{in} та P_{out}. P_{in} – множина вхідних ребер з Edges та вхідних портів з $IO_{d,in} \bigcup IO_{c,in}$. P_{out} – множина вихідних ребер з Edges та вихідних портів з $IO_{d,out} \bigcup IO_{c,out}$. У свою чергу, кожна з цих двох множин складається з двох підмножин: $P_{in} = PC_{in} \bigcup PD_{in}$, $P_{out} = PC_{out} \bigcup PD_{out}$. PC_{in} – вхідні управляючі ребра (порти), PC_{out} – вихідні управляючі ребра (порти), PD_{in} – вхідні ребра (порти) даних, PD_{out} – вихідні ребра (порти) даних.

Вхідним ребром для вершини з Nodes називається таке ребро, для якого ця вершина є вхідною. Аналогічно для вихідного ребра.

Кожна вершина $nd \in Nodes$ являє собою скінчений комбінаційний автомат, тобто автомат, який не має внутрішньої пам'яті, і вихідні сигнали в кожен момент дискретного часу встановлюються тільки на основі вхідних сигналів. В наших позначеннях вершина $nd \in Nodes$ кожен момент дискретного часу τ використовує вихідні канали елементів множини P_{in} як вхідні дані. Застосувавши функції виходів, подає числові значення на свої вихідні канали, тобто на вхідні канали елементів множини P_{out}. Відносно вершини ребра мають скорочене позначення

$$set : \overset{mdf}{\rightarrow} Node \in P_{in} , \; set : Node \overset{mdf}{\rightarrow} \in P_{out} .$$

Тут $Node$ – вершина на іншому кінці графа.

В графі QSM є множина ребер, які мають одну і ту ж вершину в ролі як вхідної, так і вихідної (петля). Наприклад, $Ed_c : Quantum_k \rightarrow Quantum_k$ використовується для збереження загального

рівня засвоєння кванта в результаті виконання різних операцій. Відносно вершин такі ребра позначаються $set : \overset{mdf}{\rightarrow}$.

Загальну схему вершини графа QSM як структурного автомата можна зобразити так (рис. 3):

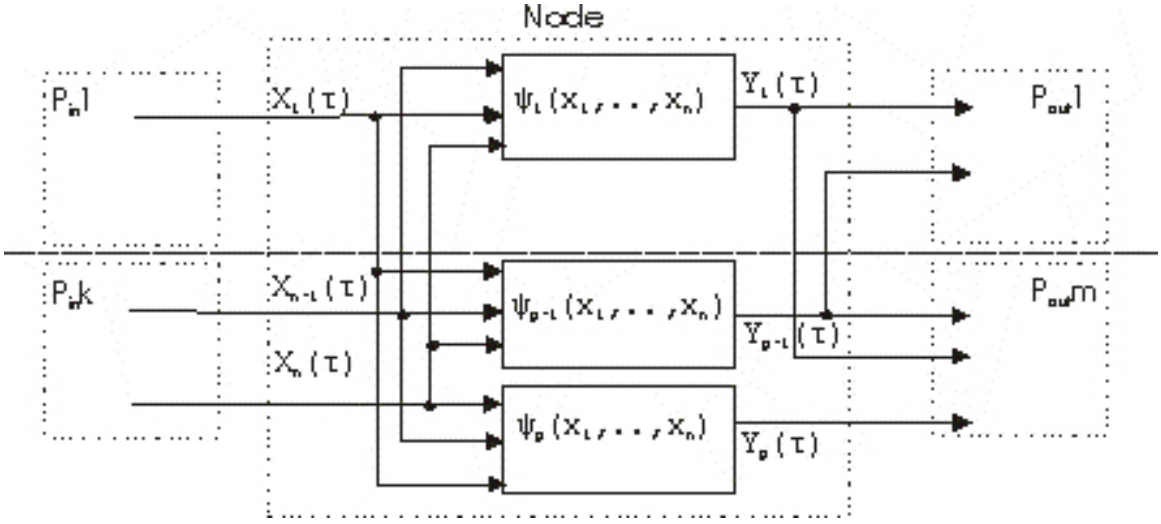


Рис. 3. Загальна схема вершини графа QSM

Кількість вхідних та вихідних каналів, вирази функцій виходів залежать від призначення вершини. Більшість аналітичних виразів для функцій $\psi_1,...,\psi_p$ є складеними функціями (значення залежить від виконання логічних умов).

4. Процес керування потоком квантів знань на основі графо-автоматної моделі

Вершини (Nodes) розділено на такі групи (індексовані підмножини): початок курсу (Start), кінець курсу (Finish), заняття (Lesson), навчальні блоки (Block), сортувачі блоків (Sorter), операції (Operation), кванти (Quantum), виконувач операцій (Operator), опрацювання квантів (Teacher).

Кожний з типів вершин має свій набір вхідних і вихідних ребер і свої функції переходу.

Кількість N вершин Lesson дорівнює кількості занять у курсі. Вершин Sorter є така ж кількість N. Вершині $Lesson_i$ відповідає вершина $Sorter_i$, $i = 1..N$.

M_i – кількість навчальних блоків i-го заняття.

K – кількість квантів всього курсу.

L – кількість операцій.

$Block_{i,j}$ – j-й навчальній блок і-го заняття.

$O_k, k = 1..K$ – кількість операцій, що використовують квант (k).

$Q_l, l = 1..L$ – кількість квантів, що застосовуються в операції (l).

Кожний тип вершин має свою структуру автомата і специфічні вирази функцій виходів. Охарактеризуємо автомат, який використовується для вершин типу $Sorter$. Вершини цього типу призначені для запуску на вивчення одного з навчальних блоків уроку при відповідному запиті.

Наступний навчальний блок визначається в залежності від його попереднього рівня засвоєння. Вершина $Sorter_i$ зв'язується з іншими вершинами за допомогою одного ребра $Ed_c:\text{Lesson}_{\text{i-1}} \rightarrow \text{Sorter}_{\text{i}}, i>1$ та по M_i ребер $Ed_c:Sorter_i \rightarrow Block_{i,j}$, $Ed_c:\text{Block}_{\text{i,j}} \rightarrow \text{Sorter}_{\text{i}}$, $Ed_d:\text{Block}_{\text{i,j}} \rightarrow \text{Sorter}_{\text{i}}$, $j=1..M_i$. Тобто вершина-автомат $Sorter$ має $4M_i+2$ входів та $3M_i+1$ виходів, де $Ed_c:Sorter_i \rightarrow Block_{i,j}$ – ребро, яке використовується для сигналізації про необхідність вивчення блоку; $Ed_c:\text{Lesson}_{\text{i-1}} \rightarrow \text{Sorter}_{\text{i}}, i>1$ – для активації наступного заняття, $Ed_d:\text{Block}_{\text{i,j}} \rightarrow \text{Sorter}_{\text{i}}$ – для передачі рівня засвоєння блоку; $Ed_c:\text{Block}_{\text{i,j}} \rightarrow \text{Sorter}_{\text{i}}$ – для сигналізації про засвоєння блоку. Задамо відповідності між вхідними та вихідними каналами вершини $Sorter$ та її ребер.

$[Ed_c:\text{Lesson}_{\text{i-1}} \rightarrow].Y_1 \rightarrow Xl$, $Yl \rightarrow [Ed_c:\text{Lesson}_{\text{i-1}} \rightarrow].X_2$, $Yb_j \rightarrow [Ed_c:\rightarrow Block_{i,j}].X_1$, $[Ed_d:Block_{i,j} \rightarrow].Y \rightarrow Xr_j$, $[Ed_c:Block_{i,j} \rightarrow].Y_1 \rightarrow Xb_j$, $Yd_j \rightarrow [Ed_c:Sorter_i \rightarrow].X_2$.

Функції виходів вершини $Sorter$ поділяються на декілька груп:

1) $Yl=\psi_l(Xl)=Xl$ – ребро $Ed_c:\text{Lesson}_{\text{i-1}} \rightarrow \text{Sorter}_{\text{i}}, i>1$ (або $Ed_c:Start \rightarrow Sorter_1$) деактивується в наступний момент часу після активації;

2) $Yb_j=\psi_{b_j}(Xl, Xr_1,..,Xr_{M_i}, Xb_1,..,Xb_{M_i})=$

$$\begin{cases} 1, & \sum\limits_{k=1..M_i} Xb_k < M_i \wedge (Xl=1 \vee \min\limits_{k=1..M_i, Xb_k=1} Xt_k=1) \wedge Xr_j = \min\limits_{k=1..M_i, Xb_k=1}(Xr_k) \wedge \sum\limits_{k=1..j, Xb_k=1, Xr_j=Xr_k} Xb_k = 0 \\ 0, & \textit{в інших випадках} \end{cases}$$

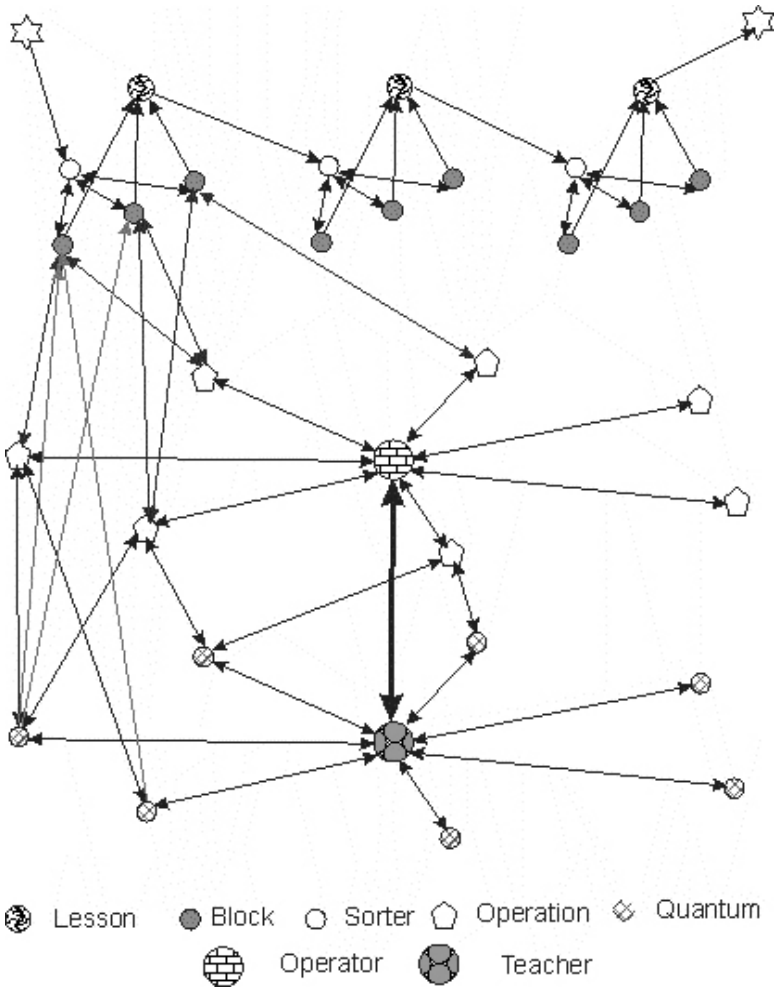
– активується одне із ребер $Ed_c:Sorter_i \rightarrow Block_{i,j}$, в залежності від рівня засвоєння відповідного навчального блоку в порівнянні з іншими блоками. Якщо всі блоки засвоєні, вершина $Sorter$ припиняє активацію вихідних ребер;

3) $Yd_j=\psi_{d_j}(Xb_1,..,Xb_{M_i})=\begin{cases} 1 & \sum\limits_{k=1}^{M_i} Xb_k = M_i \\ 0 & \sum\limits_{k=1}^{M_i} Xb_k < M_i \end{cases}$ – якщо всі блоки вивчені, ребра $Ed_c:\text{Block}_{\text{i,j}} \rightarrow \text{Sorter}_{\text{i}}$ деактивуються.

5. Загальний алгоритм роботи графа QSM

Робота починається із зовнішньої команди на вершині Start, після чого активується ребро $Ed_c:Start \rightarrow Sorter_1$. Кожна вершина типу $Sorter$ по черзі запускає на вивчення один із навчальних блоків заняття $(Block)$ в залежності від попереднього рівня засвоєння блоку. При засвоєнні блоку студентом активується інший, поки не будуть засвоєні всі блоки заняття. В цьому

випадку виконується одна із команд функції $\psi(Lesson_i)$ і наступна вершина $Sorter$ отримує сигнал про початок наступного заняття за допомогою ребра $Ed_c : \mathrm{Lesson}_{i-1} \rightarrow \mathrm{Sorter}_i, i > 1.$



Початок вивчення блоку являє собою сигнал всім операціям ($Operation$) блоку про необхідність їх опрацювання (рис. 4). Операції, у свою чергу, сигналізують потрібним квантам про необхідність їх вивчення. Кожен квант стає в чергу на вивчення, давши відповідний сигнал вершині-вчителю ($Teacher$). При цьому за допомогою ребер даних до вчителя передається додаткова інформація про квант. Це такі дані, як рівень очікуваності кванта операціями (пріоритет), час попередніх вивчень кванта і рівень його засвоєння. Ці дані впливають на вибір вчителем із черги наступного кванта для його опрацювання. Коли опрацювання кванта завершене вчителем, відповідна вершина отримує про це сигнал.

Рис. 4. Представлення алгоритму роботи графа QSM

Якщо всі кванти однієї із операцій вивчені, операція ставиться в чергу до оператора на її виконання. Пріоритет операції в черзі визначається часом простою відповідних їй квантів від моменту вивчення. При готовності операції до виконання вивчення нових квантів припиняється. Тобто виконання операцій має вищий пріоритет перед вивченням нових квантів. Це необхідно для того, щоб запобігти «забуванню» уже вивчених квантів. У графі QSM для синхронізації роботи вчителя ($Teacher$) та оператора ($Operator$) використовуються спеціальні ребра, що зв'язують ці дві вершини.

Після того, як всі операції блоку виконані, блок вважається вивченим. Рівень засвоєння блоку визначається із даних, отриманих під час виконання операцій.

Коли вивчене останнє заняття, тоді активується вершина $Finish$, і навчальний курс завершується.

Зупинимося детальніше на роботі вершин $Teacher$ та $Operator$.

Вершина $Teacher$ використовує додаткову числову функцію G. Ця функція визначає пріоритет кванта на основі поданих ним додаткових даних. Аналітичний вираз функції G визначається емпірично через практичні дослідження.

Активна робота вершини $Teacher$ починається, коли один або декілька квантів стають в чергу на засвоєння – активують відповідне ребро $Ed_c : \mathrm{Quantum}_1 \to \mathrm{Teacher}$. $Teacher$ вибирає квант, який відповідає найбільш очікуваним операціями і повідомляє зовнішньому інтерфейсу користувача про необхідність відображення даного кванта на вивчення. Функції виходу, які використовуються для цієї операції, є досить громіздкими. При запуску кванта на засвоєння враховується стан вершини $Operator$. Опрацювання квантів можливе тоді, коли не виконується ніяка операція. Після того, як квант опрацьований, ззовні буде активовано відповідне ребро (порт) з множини P_{in}. У цьому випадку автомат $Teacher$ повідомляє вершині відповідного кванта про його опрацювання і вказує час, протягом якого опрацьовувався квант.

Робота вершини $Teacher$ проводиться в синхронізації з роботою вершини $Operator$. Для цього використовується спеціальний набір ребер.

Схема автомата для вершини $Operator$ схожа до схеми $Teacher$. Різниця полягає в засобах синхронізації роботи.

Слід зауважити, що тут описано простіший варіант виводу квантів на опрацювання. У більш складному можливе одночасне виведення декількох готових до опрацювання квантів. У цьому випадку мають бути встановлені відношення між групами квантів та інформаційними блоками, що містять ці кванти. Наприклад, до опрацювання готові одночасно 3 кванти. Якщо в базі знань є об'єкт (текстовий, графічний, мультимедіа), який містить дані 3 кванти, то цей об'єкт буде відображений студенту. У випадку, коли такого об'єкта немає, кванти будуть виведені по одному чи два відразу, а пізніше – один.

Для опису такої моделі в граф QSM потрібно ввести додаткову множину вершин в ролі інформаційних блоків та ребер, які будуть задавати вміст квантів у цих блоках.

6. Висновок

Описана модель, представлена за допомогою графа QSM, вирішує задачу оптимального управління потоком квантів знань в адаптивній системі дистанційного навчання та контролю знань у процесі вивчення курсу. До переваг описаної моделі можна віднести:

- збереження логічної моделі навчального курсу в структурі графа QSM;
- інтерфейсна частина навчальної системи не залежить від конкретного курсу і є периферійним модулем ядра курсу, описаного графом QSM;
- забезпечення оптимального чергування між процесами виконання операцій та засвоєння нових квантів знань;
- вибір найбільш прийнятної послідовності видачі квантів знань для засвоєння;

- технічна реалізація моделі графа QSM для різних курсів відрізняється лише кількістю елементів (вершин, ребер) та попарними зв'язками між ними, але не будовою елементів;
- модель враховує можливість «забування» квантів знань студентом;
- враховується повторюваність одних і тих же квантів та операцій у різних заняттях і контекстах.

Описана модель управління потоком квантів дозволяє реалізувати в системі дистанційного навчання і контролю знань значні можливості адаптації до особливостей сприйняття і рівнів знань тих, хто навчається.

СПИСОК ЛІТЕРАТУРИ

1. Дистервег А. Избранные педагогические сочинения. – М., 1936. – С.118.
2. Федорук П.І. Технологія розробки навчального модуля в адаптивній системі дистанційного навчання та контролю знань // Математичні машини і системи. – 2005. – № 3. – С.155–165.
3. Сирожа И.Б. Квантовые модели и методы инженерии знаний в задачах искусственного интеллекта // Искусственный интеллект. Научно-технический журнал. – 2002. – № 3. – С.161–171.
4. Аднан Али Иерархическая формальная модель дедуктивных запросов // Математические машины и системы. – 1999. – № 2. – С. 70–77.
5. Котов В.Е. Сети Петри. – М.: Наука, 1984. – С.160.
6. Вальковский В. Распараллеливание алгоритмов и программ. Структурный подход. – М.: Наука, 1989. – С.176.
7. Кристофидес Н. Теория графов. Алгоритмический подход. – М.: Мир, 1978. – С.432.
8. Пономарев В.Ф. Конечные автоматы: Учебное пособие. – Калининград: КГТУ, 1999. – С.139.