УДК 62-50

О.О. Павлюк, П.І. Бідюк

МОДЕЛЮВАННЯ ПОВЕДІНКИ КОРИСТУВАЧА МЕРЕЖІ ІНТЕРНЕТ ЗА ДОПОМОГОЮ ІЄРАРХІЧНОЇ БАЙЄСІВСЬКОЇ МОДЕЛІ

Розглянуто методику побудови ієрархічних байєсівских моделей за допомогою програмного середовища WinBugs на прикладі моделювання поведінки користувача мережі Інтернет.

Ключові слова: байєсівська мережа, моделювання поведінки, ієрархічна БМ, метод Монте-Карло.

Вступ

Байєсівські мережі широко використовуються для розв'язання задач в умовах наявності невизначеностей. У роботі [1] запропоновано розширення формалізму байєсівських мереж – ієрархічні байєсівські мережі (БМ), які можуть представляти додаткову інформацію про структуру змінних. Ієрархічні байєсівські мережі подібні до байєсівських мереж, вони представляють собою імовірнісні залежності між змінними як орієнтований ациклічний граф, де кожний вузол графа відповідає випадковій величині і кількісно визначається значеннями цієї змінної при певному значенні батьківських вузлів. Досить докладно ієрархічні байєсівські мережі розглянуті у роботі [2]. Фактично, ієрархічна байєсівська мережа є узагальненням статичної БМ, у якій вузол у мережі може бути представлено змішаним типом даних. Це дає можливість випадковим величинам мережі представляти довільно структуровані типи. В одному вузлі також можуть бути зв'язки між компонентами, що виражають імовірнісні залежності між частинами конструкції. Ієрархічні байєсівські мережі (ІБМ) можуть містити глибші знання про структуру змінних і використовувати ці знання для створення реалістичнішої ймовірнісної моделі. Згідно з роботою [3], ієрархічна байєсівська мережа складається з двох частин. Структурна частина містить змінні мережі і описує відношення та імовірнісні залежності між ними. Структурні відношення можуть бути проілюстровані або у вигляді вкладених вузлів, або у вигляді дерева ієрархії. Друга частина ієрархічної БМ – імовірнісна, вона складається з таблиць умовних ймовірностей, які кількісно характеризують представлення структурної частини.

 ISSN 1562-9945

Дослідження останніх років свідчать про поширення застосування ієрархічних байєсівських мереж. Вони використовуються при розв'язанні задач прогнозування, класифікації, моделювання процесів довільної природи та інших. Так, у роботі [4] наводиться приклад використання ієрархічної байєсівської мережі для визначення руху частин тіл взаємодіючих суб'єктів. Задача розпізнавання ґрунтується на алгоритмах обробки спостережень, які включають процедури сегментації та дослідження важливих ділянок зображення і визначення особливостей об'єкта. У цій роботі запропонована нова ієрархічна структура для визначення докладної взаємодії двох суб'єктів з використанням ієрархічної байєсівської мережі. У роботі [5] розглядається подібна задача – моделювання руху правої руки віолончеліста під час гри. Також представлено метод побудови ІБМ шляхом використання алгоритму формування стохастичної вибірки та метод навчання ІБМ на основі міри структурної подібності Купера і Гершковича. У роботі [6] розглянуто застосування ІБМ для автоматичної класифікації текстової інформації, виконується класифікація тестового набору текстів Reutars-21578 (збірка новин). Після визначення структури використовується алгоритм максимізації математичного сподівання (EM-алгоритм) для визначення параметрів класифікатора.

Дана робота присвячена розв'язанню задачі практичної реалізації методів Монте-Карло для марковських ланцюгів у програмному середовищі WinBUGS. WinBUGS – це інтерактивна версія програми Bugs для Windows. Ця програма призначена для байєсівського аналізу комплексних статистичних моделей і використовує алгоритми методів Монте-Карло для марковських ланцюгів (МКМЛ) для оцінювання параметрів моделей. WinBUGS дає можливість описувати моделі, використовуючи модифікований варіант мови BUGS, або за допомогою системи графічного представлення моделей Doodles, яке може бути трансформоване у текстовий опис. Основною перевагою мови BUGS є те, що вона гнучкіша ніж графічне представлення.

Постановка задачі

Робота присвячена застосуванню методів Монте-Карло для марковських ланцюгів до розв'язання задачі моделювання поведінки користувача мережі Інтернет на конкретному сайті з використанням програм-

ного середовища WinBugs. Поведінка користувача важлива з точки зору оцінювання ефективності дизайну сайту та оптимального ранжирування його сторінок при видачі пошукових систем. Загальним підходом до моделювання поведінки користувача при перегляді сайту є припущення, що вибір наступної сторінки – це випадковий крок з імовірнісним розподілом часу переходу на іншу сторінку; цей процес має обернений гаусівський розподіл з двома параметрами. Інший підхід до моделювання цього процесу ґрунтується на припущенні, що користувач на кожній сторінці виконує незалежний експеримент Бернуллі з метою прийняття рішення стосовно зупинки на цій сторінці. Такий підхід передбачає використання геометричного розподілу.

Для розв'язання задачі використовується байєсівська ієрархічна модель підрахунку кількості відвіданих користувачем сторінок для того щоб отримати апостеріорний розподіл частоти відвідання сторінки. Це дає можливість згрупувати (кластеризувати) сесії користувачів у порівняно невелику кількість груп. Модель повинна мати достатню кількість параметрів для того щоб відповідати даним і при цьому використовувати такий розподіл даних, який дасть можливість структурувати залежність між параметрами. Вона може бути узагальнена для різних типів веб-сайтів на різних рівнях агрегації сторінок і різних схем кластеризації.

Основна мета роботи – розробка методики застосування методів МКМЛ до аналізу та моделювання поведінки користувача у мережі Інтернет при перегляді вебсайту на основі даних відкритої статистики відвідування вебсайтів.

Особливості структури байєсівських ієрархічних моделей

Ієрархічні байєсівські мережі – це узагальнені БМ, у яких вузол мережі може мати складну структуру. Це дозволяє випадковим змінним мережі представляти структури довільного типу. В одному вузлі можуть бути і зв'язки між компонентами, що представляють ймовірнісну залежність між частинами структури. Ієрархічні байєсівські моделі задають умовні ймовірнісні залежності так само, як звичайні байєсівські мережі. ІБМ можуть відображати глибші знання стосовно структури змінних і використовувати ці знання для створення реалістичних імовірнісних моделей. Такі мережі складаються з двох частин – структурної та імовірнісної.

Наведемо формальне означення поняття ІБМ. Введемо ієрархічні складові, на яких визначається мережа. В якості агрегатної функції скористаємось декартовим добутком множин.

Елементарний тип даних – це певний набір констант. Якщо ми маємо $\{\tau_1, \tau_2, \ldots, \tau_n\}$ певну кількість елементарних типів, то декартів добуток $\tau = \tau_1 \times \tau_2 \times \ldots \times \tau_n$ є складеним типом; типи даних $\tau_1, \tau_2, \ldots, \tau_n$ – це компоненти типу τ.

Структурований тип, що відповідає типу τ – це дерево t, яке має такі властивості:

– якщо τ – елементарний тип, тоді t – це єдиний вузол під назвою τ;

– якщо τ – складений тип, тоді t має корінь τ, а в якості нащадків структуровані типи компонентів τ.

Структура дерева ієрархічної байєсівської мережі T над структурованим типом t – це набір $< R, V, E >$, де:

– R – корінь структури, що відповідає випадковій величині типу τ;

– V – набір структур дерев над типами-компонентами τ (якщо τ – елементарний тип, то множина V - порожня);

– $E \subset V^2$ – множина спрямованих зв'язків між елементами V, таких, що результуючий граф не містить спрямованих циклів.

Імовірнісна частина ієрархічної байєсівської мережі, пов'язана із структурою T, складається з таких елементів:

1) таблиця ймовірностей для кожної змінної з T, що не має батьківських вузлів;

2) таблиця умовних ймовірностей інших змінних із урахуванням значень батьківських вузлів.

З урахуванням наведених означень, можна сказати, що ієрархічна байєсівська мережа це трійка $< T, P, t >$, де: t – це структурований тип даних; $T =< R, V, E >$ – структура дерева ієрархічної байєсівської мережі на множині t; P – це імовірнісна складова ієрархічної байєсівської мережі.

Особливості оцінювання байєсівських ієрархічних моделей

ІБМ можна застосовувати для формування імовірнісного виводу стосовно змінних, які вони містять. Одним з ефективних методів отри-

мання цього висновку є застосування методів Монте-Карло для ланцюгів Маркова (МКМЛ). Розглянемо цю групу методів докладніше.

Метод Монте-Карло для ланцюгів Маркова – це клас алгоритмів для вибірки з імовірнісних розподілів, заснований на побудові ланцюга Маркова, який має необхідний розподіл, у якості рівноважного розподілу. Після достатньої кількості кроків стан ланцюга береться як вибірка з досліджуваного розподілу. Метод грунтується на генеруванні значень певної випадкової величини θ, математичне сподівання якої представляє досліджувану нами величину. Потім отримані значення коригуються щоб досягти кращої відповідності апостеріорному розподілу $p(\theta|y)$. Вибірки генеруються послідовно і поточні значення однієї вибірки залежать лише від попереднього значення.

Моделювання ланцюга Маркова використовується тоді, коли неможливо (або неефективно) генерувати θ безпосередньо з розподілу $p(\theta|y)$. Замість цього ітераційно генерується вибірка таким чином, що на кожному етапі цього процесу її розподіл наближається до $p(\theta|y)$. Для широкого класу задач (у тому числі для обчислення апостеріорного розподілу для багатьох ієрархічних моделей) цей підхід є відносно простим способом отримати надійні результати. Ключовим моментом при моделюванні ланцюга Маркова є створення марковського процесу, стаціонарним (усталеним) станом якого буде бажаний розподіл $p(\theta|y)$. Після побудови моделі і генерування вибірки необхідно перевірити збіжність змодельованих послідовностей.

Багато методів Монте-Карло для ланцюгів Маркова використовують модель випадкового кроку, вони рухаються в околі стаціонарного розподілу відносно невеликими кроками, без вибору певного напрямку. Ці методи прості стосовно реалізації та аналізу, але можуть вимагати багато часу для реалізації. Найбільш поширеними з цих методів є вибірка за Гіббсом та алгоритм Метрополіса-Гастінгса.

Вибірка Гіббса – це алгоритм моделювання ланцюга Маркова, який був визнаний корисним у багатьох багатовимірних задачах. Нехай вектор параметрів θ розділено на d компонент або підвекторів $\theta = (\theta_1, \ldots, \theta_d)$. Кожна ітерація вибірки Гіббса проходить через підвектори θ, генеруючи кожну підмножину в залежності від значень всіх

інших підмножин. Таким чином на кожній ітерації t виконується d кроків. На кожній ітерації t кожне значення θ_j^t вибирається з умовного розподілу із урахуванням всіх інших компонентів θ:

$$p(\theta_j \left| \theta_{-j}^{t-1}, y \right.), \tag{1}$$

де θ_{-j}^{t-1} – всі компоненти θ, за виключенням θ_j. Таким чином, кожний підвектор θ_j оновлюється в залежності від решти значень елементів θ.

Алгоритм Метрополіса – це адаптація моделі випадкового блукання, що використовує правило прийняття або відхилення випадкового значення для того щоб наблизитись до заданого розподілу. Алгоритм працює таким чином:

1) Вибираємо відправну точку θ^0, для якої $p(\theta^0 \left| y \right.) > 0$, з початкового розподілу $p_0(\theta)$.

2) Для $t = 1,2,\ldots$ виконуємо такі кроки.

а) Генеруємо значення θ^* з перехідного розподілу у момент часу t, $J_t(\theta^* \left| \theta^{t-1} \right.)$. Для алгоритму Метрополіса (але не Метрополіса-Гастінгса) перехідний розподіл повинен бути симетричним, тобто задовольняти умові $J_t(\theta_a \left| \theta_b \right.) = J_t(\theta_b \left| \theta_a \right.)$ для всіх θ_a, θ_b, і t.

б) Розраховуємо відношення:

$$r = \frac{p(\theta^* \left| y \right.)}{p(\theta^{t-1} \left| y \right.)}. \tag{2}$$

в) Покладаємо:

$$\theta^t = \begin{cases} \theta^*, при & \min(r,1) \\ \theta^{t-1}, інакше \end{cases}. \tag{3}$$

З огляду на поточне значення θ^{t-1}, перехідний розподіл $T_t(\theta^t \left| \theta^{t-1} \right.)$ ланцюга Маркова є сумішшю сталої точки $\theta^t = \theta^{t-1}$ і зваженої версії перехідного розподілу $J_t(\theta^t \left| \theta^{t-1} \right.)$.

Алгоритм Метрополіса-Гастінгса узагальнює основний алгоритм Метрополіса. По-перше, правило переходу J_t не має бути обов'язково симетричним, тобто не існує вимоги, що $J_t(\theta_a|\theta_b) = J_t(\theta_b|\theta_a)$. По-друге, для корекції асиметрії співвідношення r замінюється на співвідношення коефіцієнтів:

$$r = \frac{p(\theta^*|y) / J_t(\theta^*|\theta^{t-1})}{p(\theta^{t-1}|y) / J_t(\theta^{t-1}|\theta^*)}. \quad (4)$$

Завдяки асиметрії у правилі переходу підвищується швидкість реалізації процесу випадкового кроку.

Побудова моделі відвідування порталу новин

Розглянемо модель для порталу новин msnbc.com. Сторінки сайту згруповані у 17 категорій: 1 = "Головна", 2 = "Новини", 3 = "Технології", 4 = "Місцеві новини", 5 = "Думка", 6 = "Онлайн", 7 = "Різне", 8 = "Погода", 9 = "Здоров'я", 10 = "Життя", 11 = "Бізнес", 12 = "Спорт", 13 = "Підсумки", 14 = "BBS (дошка оголошень)", 15 = "Подорожі", 16 = "MSN-новини" і 17 = "MSN-спорт". Кількість адрес сторінок у кожній категорії коливається від 10 до 5000. Данні, що використовуються в даній роботі, отриманні з міжнародного інформаційного сервісу (IIS) – це журнали відвідування сайту за один день 28 вересня 1999 року. Приклад даних:

Користувач 1: 1, 1

Користувач 2: 2

Користувач 3: 3, 2, 2, 4, 2, 2, 2, 3, 3

Це означає, що перший користувач відкрив msnbc.com через головну сторінку і натиснув там посилання. Користувач 3, навпаки, зайшов на сайт через сторінку "Технології", перейшов до "Новин" і переглянув 2 сторінки в цій категорії, потім до "Місцевих новин", звідти повернувся до "Новин" і відвідав там 3 сторінки і ще 2 сторінки в розділі "Технології".

Структура сайту дозволяє користувачеві вільно переходити від однієї групи до іншої, тобто порядком перегляду сторінок можна знехтувати. На основі наведених даних можна побудувати модель:

– користувач 1: 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2;

– користувач 2: 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1;

– користувач 3: 0 5 3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3.

Перші 17 стовпчиків – це тривалість перебування користувача в кожній з 17 категорій; фактично це кількість сторінок, відвіданих у кожній категорії. Змінна 18 має два можливих значення: – 1, якщо користувач зайшов на сайт через головну сторінку, і – 0, якщо через іншу сторінку. У стовпчику 19 відображена загальна глибина сесії, тобто кількість різних категорій що були відвідані користувачем.

Тепер можна побудувати статистичну модель, у якій одиницею спостереження є сесії відвідувача сайту msnbc.com. Кожному сеансу користувача відповідає вектор вимірів $y_i = [y_{i1}, y_{i2}, \ldots, y_{i17}]$. Кожний вимір y_{ij} визначає скільки разів користувач i відвідав сторінки j-ї категорії веб-сайту msnbc.com. Випадкова змінна y_{ij} розподілена за законом Пуассона, з параметром μ_{ij}(математичне сподівання змінної y_{ij}), яке залежить від ряду детермінованих (статистичних) та випадкових факторів.

Результати, які містяться у даних, занадто неоднорідні і не дають можливості припустити, що модель – це звичайний пуасонівський розподіл. Це приводить до висновку, що відмінності між сесіями користувачів зумовлені не тільки фактором випадковості. Тобто наявне підвищене розсіювання внаслідок системних (невипадкових) факторів, які необхідно врахувати у моделі. Тому доцільно розглянути загальну лінійну змішану модель (Generalized Linear Mixed Model (GLIMM)) з випадковими факторами для кожного сеансу роботи користувача.

У моделі, яка використовується для опису відвідувань сайту msnbc.com, не всі фактори випадкові, тобто у модель необхідно ввести статичні і випадкові змінні:

$$\log(\mu_{ij}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + b1_i + b_{ij}, \qquad (5)$$

де: x_1 – сторінка, через яку зайшов користувач ($x_1 = 1$ для головної сторінки, і 0 – для решти); x_2 – глибина (1-17); x_3 – індикатор, який дорівнює 1 для головної сторінки, і 0 – для решти сторінок. Це зумовлено важливою роллю, яку відіграє головна сторінка і тим фактом, що вона є найбільш відвідуваною, і саме через неї заходять на сайт більшість користувачів. Інші змінні такі: $b1_i$ – випадкова змінна, що відображає до-

даткові відмінності між сеансами користувачів; b_{ij} – випадкова змінна для моделювання випадкового розподілу всередині одного сеансу.

Єдина відмінність між статичними і випадковими параметрами полягає у тому, що коваріації статичних параметрів – це константи, в той час як коваріації випадкових ефектів залежать від невідомих гіперпараметрів $\theta = (\tau_{b1} + \tau_b)$, які необхідно оцінити за наявними даними. Отже потрібно задати апріорний розподіл θ. Це приводить до того, що модель буде ієрархічною. Таким чином, повна модель має вигляд:

$$y_{ij} = \frac{\mu_{ij}^{y}}{y_{ij}!} e^{-\mu_{ij}}; \tag{6}$$

$$\log(\mu_{ij}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + b1_i + b_{ij}. \tag{7}$$

Для того щоб апріорні розподіли статичних ефектів були неінформативними, їм задається нормальний апріорний розподіл із значеннями параметрів, дуже близьких до 0 […]:

$$\beta_i \sim N(0{,}0;\ 0{,}0001). \tag{8}$$

Апріорні розподіли для випадкових ефектів:

$$b1_i \sim N(0{,}0;\ \tau_{b1}), \tag{9}$$

$$b_{ij} \sim N(0{,}0;\ \tau_b). \tag{10}$$

Для τ_{b1} і τ_b також задаються неінформативні апріорні гамма-розподіли з дуже малими значеннями параметрів:

$$\tau_{b1} \sim \Gamma(0{,}001;\ 0{,}001); \tag{11}$$

$$\tau_b \sim \Gamma(0{,}001;\ 0{,}001); \tag{12}$$

$$\sigma_{b1} = \frac{1}{\sqrt{\tau_{b1}}};\ \sigma_b = \frac{1}{\sqrt{\tau_b}}. \tag{13}$$

Визначення моделі мовою системи WinBugs

Мова системи WinBugs дозволяє створювати короткий опис моделі за допомогою псевдокоду, використовуючи відповідні символи. Зокрема, для позначення стохастичних (ймовірнісних) відношень застосову-

ється знак ~, а для позначення детермінованих (логічних) відношень – знак лівої стрілки (знак '<' та '–').

Програмна модель даного прикладу така:

```
model
{
for( i in 1 : N ) {
for( j in 1 : 17 ) {
y[i, j] ~ dpois(m[i, j])
b[i,j] ~ dnorm(0.0, tau)
log(m[i,j]) <- beta0 + beta1 * x1[i] + beta2 * x2[j] + beta3 * x3[j] + b1[i] +b[i,j]
}
b1[i] ~ dnorm(0.0, tau1)
}
beta0 ~ dnorm(0.0, 0.0001)
beta1 ~ dnorm(0.0, 0.0001)
beta2 ~ dnorm(0.0, 0.0001)
beta3 ~ dnorm(0.00, 0.0001)
tau1 ~ dgamma(0.001, 0.001)
tau ~ dgamma(0.001, 0.001)
sigma1 <- 1 / sqrt(tau1)
sigma <- 1 / sqrt(tau)
}
}
```

Програмне забезпечення WinBUGS використовує «змішані документи», які можуть включати різноманітні типи інформації (форматовані тексти, таблиці, формули, графи та ін.), які відображаються в одному вікні і зберігаються в одному файлі. Але для зручності краще розділяти інформацію і зберігати модель і дані у різних файлах

Для створення та оцінювання моделі необхідно виконати такі кроки:

1. Створюємо файл моделі кнопками «File»->«New» та задаємо у новому вікні параметри моделі так як це показано на рис. 1.

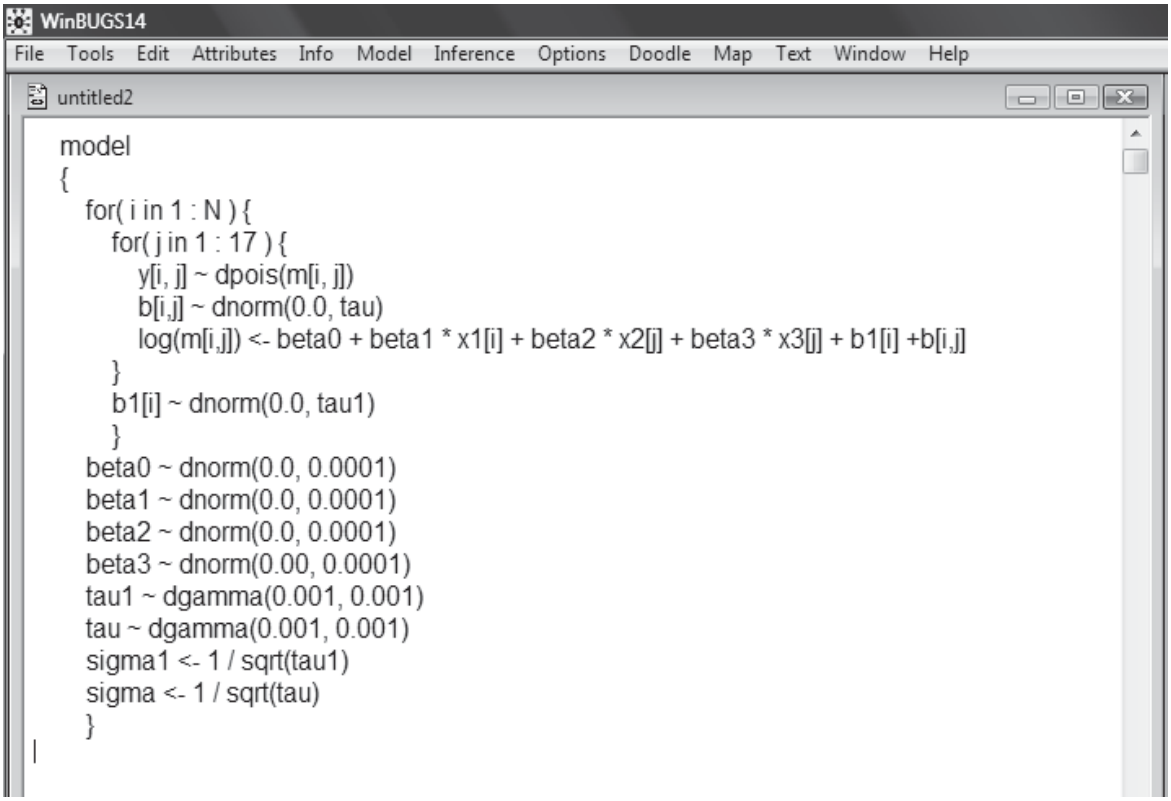

Рисунок 1 – Файл моделі

2. Перед запуском програми (що описує модель) необхідно спочатку переконатися у тому, що опис моделі повністю визначає імовірнісну модель; для цього необхідно:

– на панелі інструментів та натиснути один раз ліву кнопку миші на «Model»;

– вибрати пункт «Specification...» та натиснути один раз ліву кнопку миші;

– обрати вікно, що містить код моделі;

– виділити мишкою слово model на початку коду;

– перевірити синтаксис моделі, натиснувши один раз ліву кнопку миші на «check model» у вікні інструментарію «Specification Tool». Повідомлення з текстом "model is syntactically correct" (модель є синтаксично правильною) повинно з’явитися зліва внизу програмного вікна WinBUGS, як це показано на рис. 2:

 ISSN 1562-9945

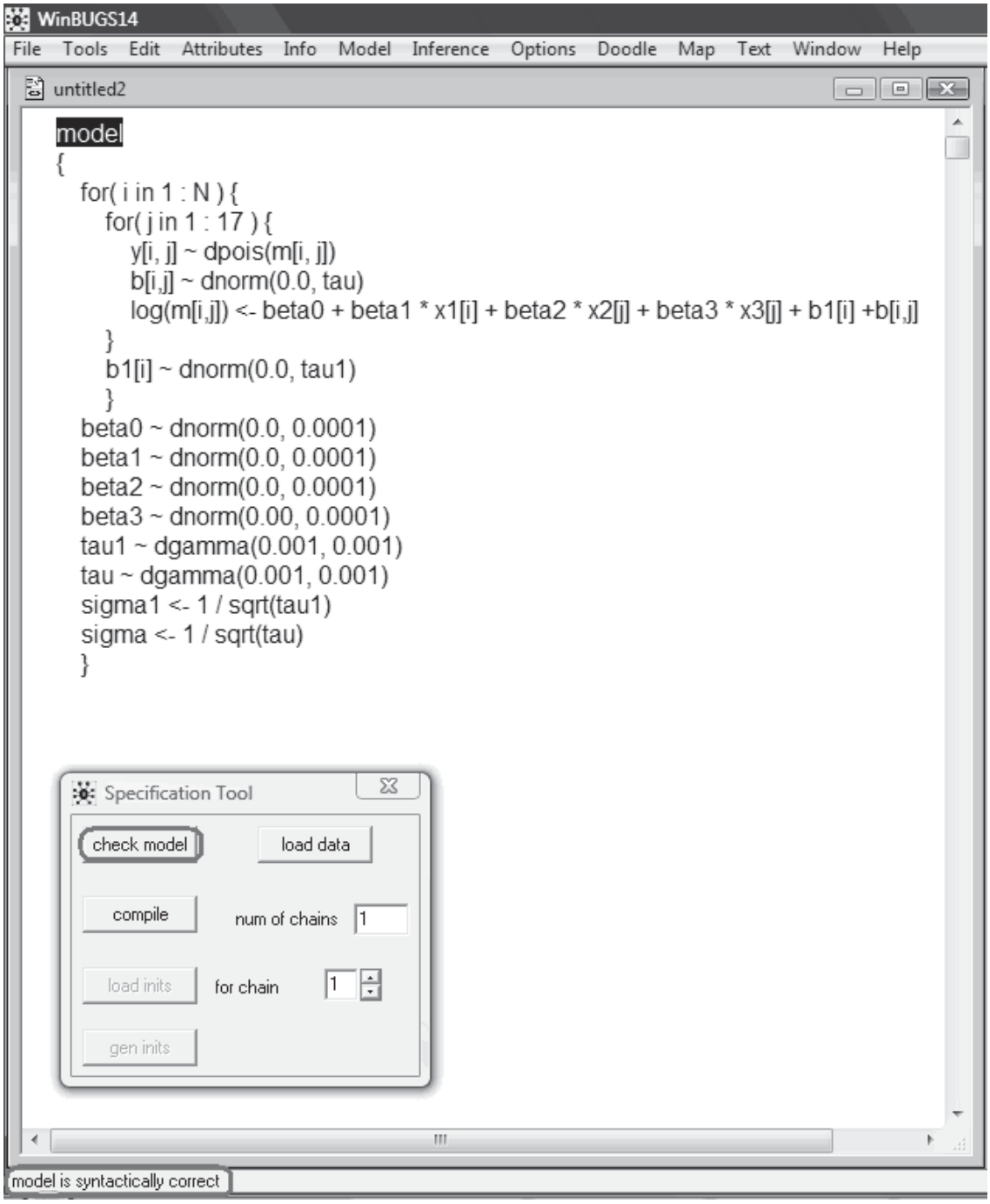

Рисунок 2 – Перевірка правильності моделі

3. Далі створюємо файл з даними. Для завантаження даних у WinBUGS використовуються файли у форматі S-Plus. Цей формат дозволяє задавати скалярні величини та масиви і присвоювати їм значення у єдиній структурі з ключовим словом «list». Приклад такого файлу наведено на рис. 3:

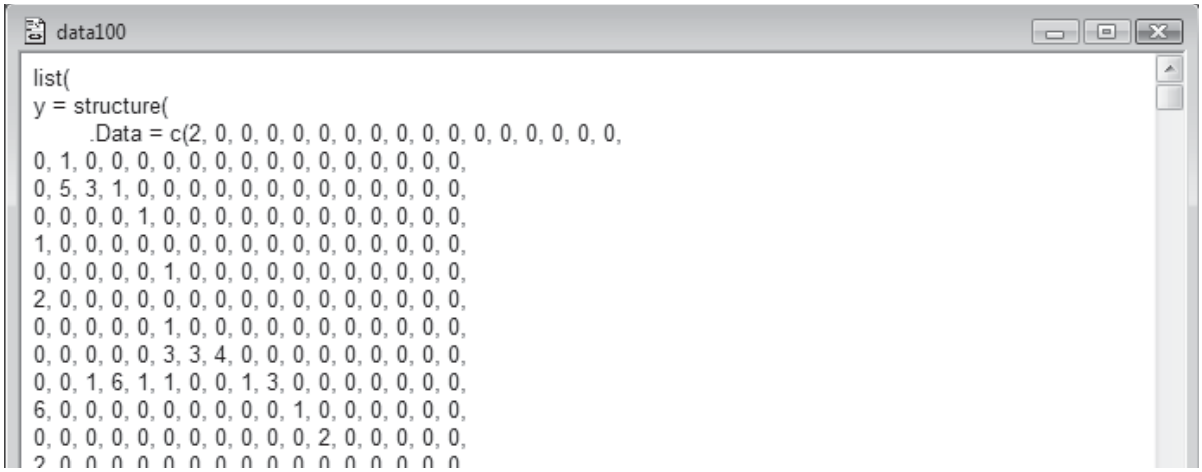

Рисунок 3 – Створюємо файл даних

У цьому файлі задаємо такі значення:

N – кількість спостережень; у даному прикладі для спрощення експерименту використаємо 100 спостережень з наявних даних;

y_{ij} – кількість відвідувань користувачем i сторінки j-ї категорії, це перші 17 стовпчиків наших даних, описаних у розділі «Побудова моделі»;

x_1 – сторінка, через яку зайшов користувач; це 18-й стовпчик даних, описаних у розділі «Побудова моделі»;

x_2 – глибина сеансу; це 19-й стовпчик даних, описаних у розділі «Побудова моделі»;

x_3 – індикатор головної сторінки (x3 = c(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)).

Для того щоб завантажити дані, необхідно виконати таку послідовність дій:

- Виділити слово «list» на початку файлу.

– Лівою кнопкою миші один раз натиснути на «load data» у вікні «Specification Tool». У лівому нижньому куті вікна програми має з'явитися повідомлення «data loaded» (дані завантажені); це показано на рис. 4.

Рисунок 4 – Завантаження даних

4. Тепер необхідно обрати кількість ланцюгів даних (тобто вибірок для моделювання). За замовчуванням ця кількість дорівнює одиниці, але у кожному окремому випадку можна обрати будь-яку кількість. У даному прикладі застосовуємо 2 ланцюги, оскільки використання декількох ланцюгів – це один із шляхів перевірки збіжності моделювання за методом MCMC. Необхідно задати 2 у полі «num of chains» вікна «Specification Tool».

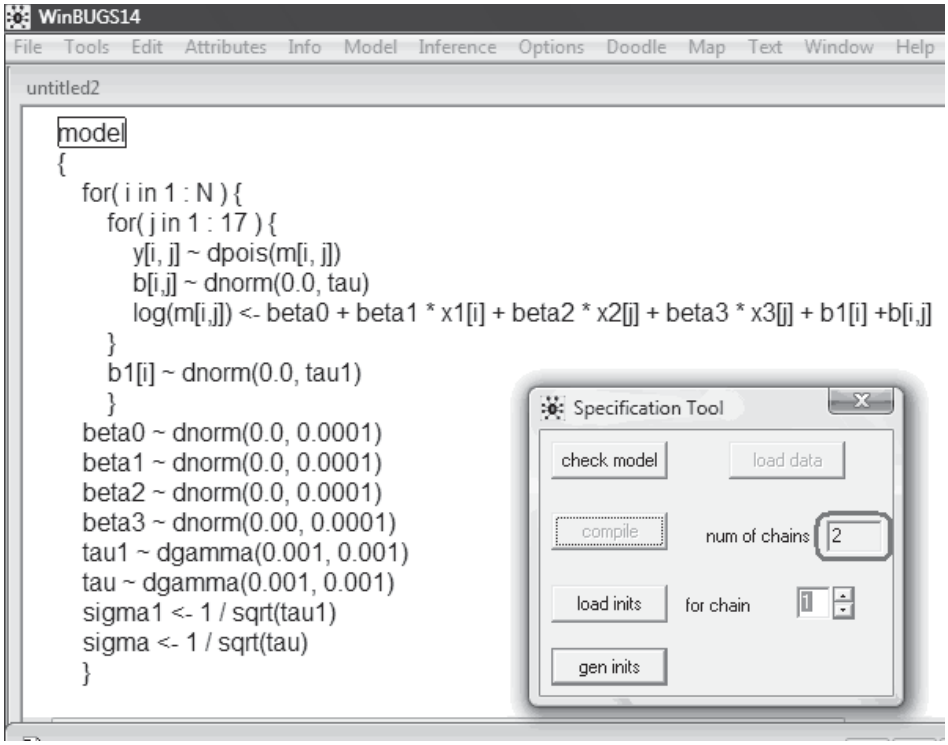

Рисунок 5 – Задаємо кількість ланцюгів

5. Далі необхідно запустити компіляцію програми. Для цього потрібно натиснути лівою кнопкою миші кнопку «compile» у вікні «Specification Tool». У нижньому лівому куті повинно з'явитися повідомлення «model compiled» (модель скомпільована). Це задає внутрішню структуру моделі та обирає спеціальні алгоритми МСМС для генерування даних.

6. Тепер потрібно задати деякі початкові значення для кожного стохастичного вузла. Це можуть бути довільні значення. Проте на практиці збіжність моделі може бути поганою якщо обрано занадто невдалі значення. Тому необхідно вказати різні набори початкових величин для кожного ланцюга, тобто для нашого випадку необхідно задати 2 набори, адже потрібно визначити два ланцюги.

Початкові дані не обов'язково задавати для всіх параметрів моделі, WinBUGS дає можливість генерувати початкові величини для кожного стохастичного параметра, який не є ініціалізованим. У даному прикладі задаємо початкові значення τ_{b1} і τ_b. Для обох ланцюгів по черзі виділяємо слово «list» відповідного набору початкових величин та натискаємо на кнопку «load inits» у вікні «Specification Tool».

Рисунок 6 – Компілюємо модель

У нижньому лівому куті з'являється повідомлення «the chain contains uninitialized nodes» (ланцюг містить неініціалізовані змінні).

7. Закриваємо вікно «Specification Tool». Тепер все підготовлене для запуску імітаційного моделювання. Але спочатку потрібно задати параметри, для яких ми бажаємо зберегти згенеровані данні. Для цього натискаємо «Inference» –> « Samples...», у вікні «Sample Monitor Tool» набираємо ім'я змінної, значення якої необхідно зберегти, та натискаємо «Set».

У даному прикладі ми задали зміні beta0, beta1, beta2, beta3, sigma та sigma1.

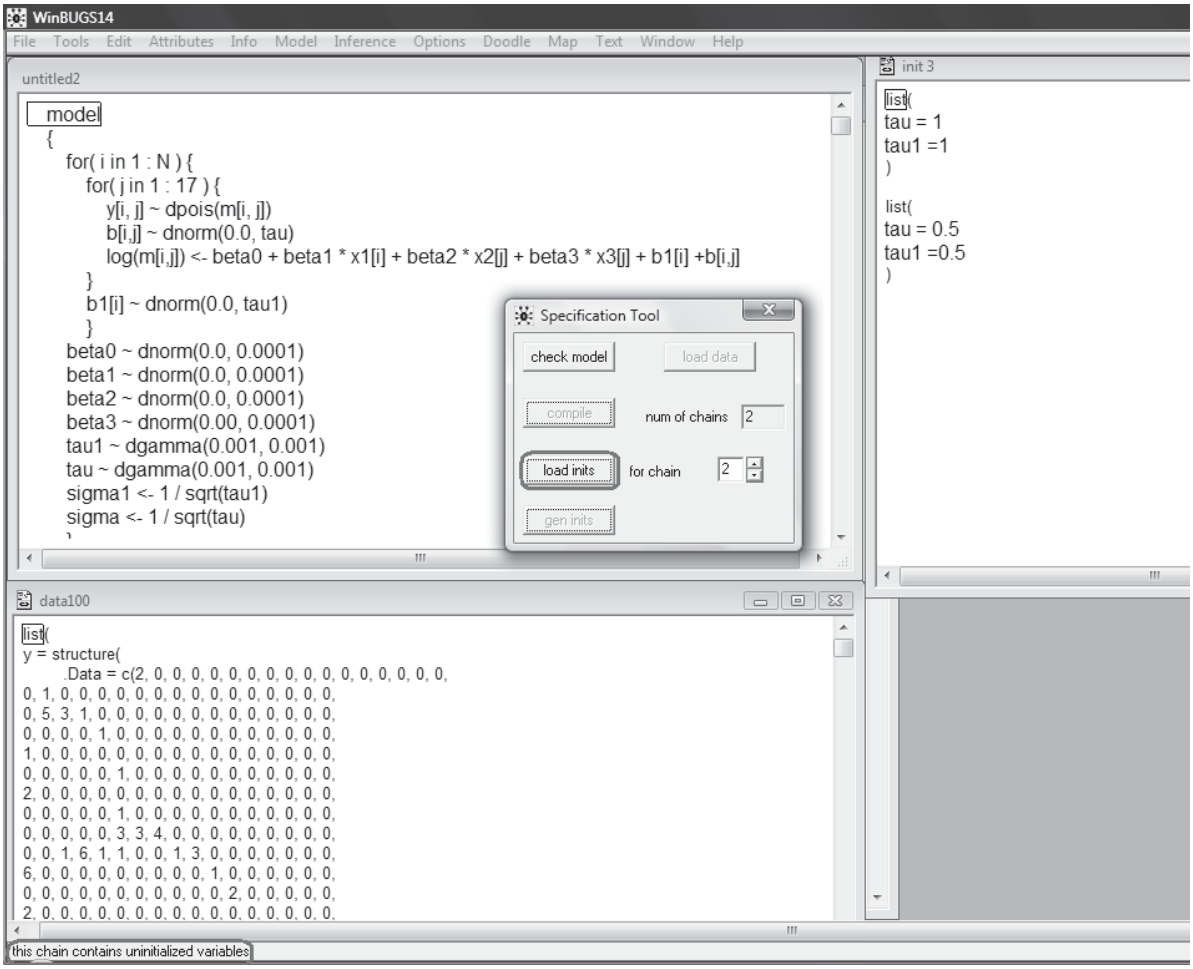

Рисунок 7 – Завантажуємо початкові данні

За допомогою кнопки «gen inits» у вікні «Specification Tool» генеруємо початкові значення для неініціалізованих змінних. У нижньому лівому куті з'являється повідомлення «initial values generated, model initialized» (початкові данні сгенеровано, модель ініціалізовано).

Для запуску процедури імітаційного моделювання необхідно:

– обрати опцію «Update...» з меню «Model»;

– ввести кількість ітерацій імітаційного моделювання (за замовчуванням дорівнює 1000);

– натиснути кнопку «update»: программа розпочинає процес моделювання величин для кожного параметра моделі; це може зайняти декілька секунд: у вікні «iteration» буде показано, скільки ітерацій було виконано.

– коли процедура моделювання закінчиться, у нижньому лівому куті програми повинно з'явитися повідомлення "updates took XXX s" (оновлення зайняло XXX секунд), де XXX – це кількість секунд, витрачених для завершення моделювання.

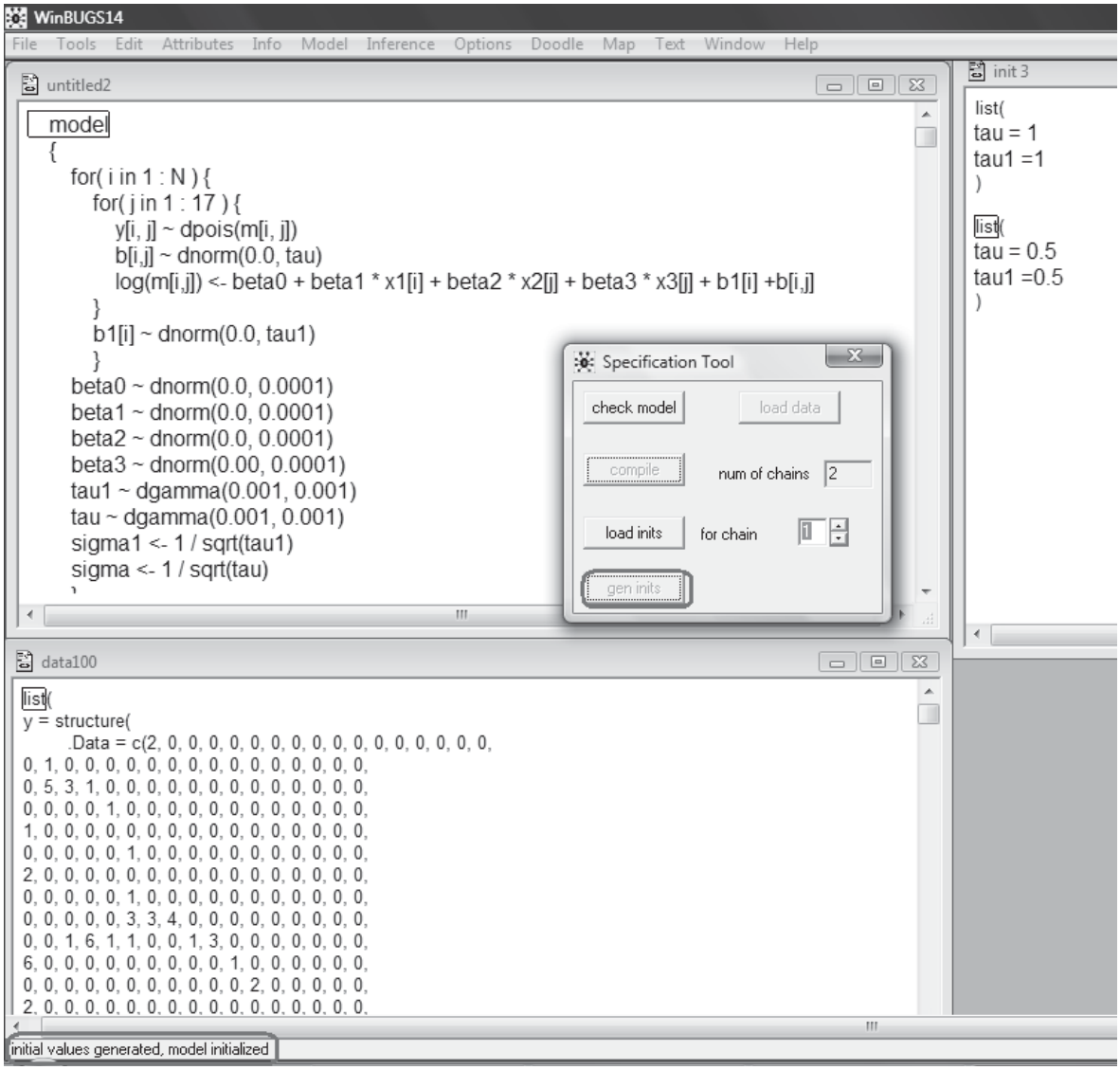

Рисунок 8 – Генерування початкових значень

Якщо попередньо встановити спостереження за декількома параметрами, можна перевірити збіжність моделі та подивитися графічні та чисельні результати симуляції. Відкривши вікно «Sample Monitor Tool» можна обрати конкретну зміну або вибрати усі набравши «*». Натиснувши кнопку «history», побачимо графіки обраних змінних і зможемо оцінити збіжність моделі.

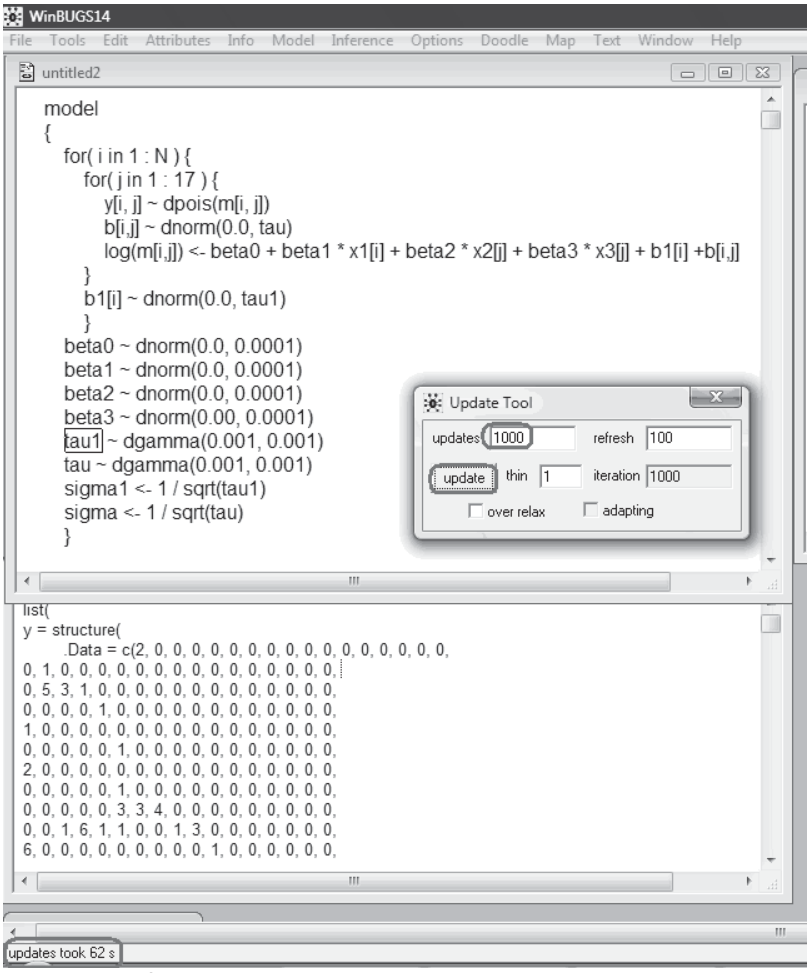

Рисунок 9 – Виконання імітаційного моделювання

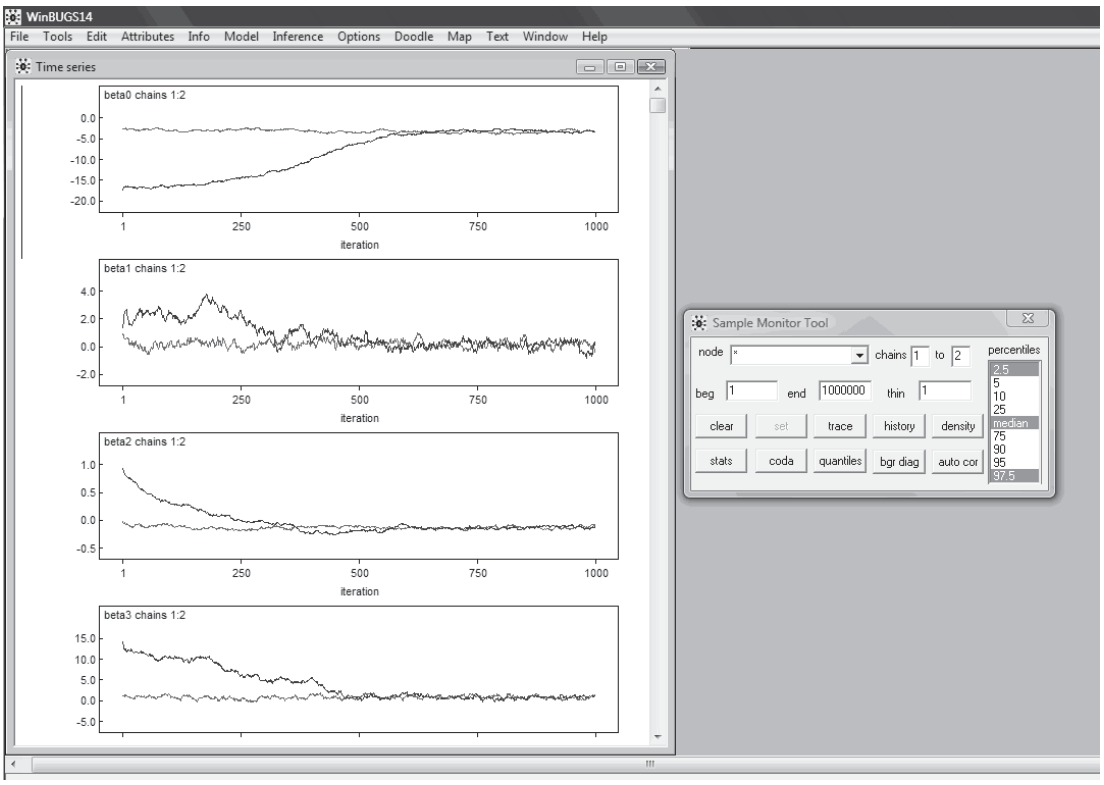

Рисунок 10 – Перевірка збіжності

Якщо результати моделювання виявилися збіжними, необхідно запустити декілька подальших оновлень для обрахунку параметрів апостеріорного розподілу.

Коли задано достатньо оновлень для спостереження параметрів апостеріорного розподілу, можна оцінити процедуру моделювання чисельно або графічно. Для отримання чисельних параметрів розподілу потрібно у вікні «Sample Monitor Tool» обрати бажану зміну або вибрати усі зміни, набравши «*», та натиснути кнопку «stats».

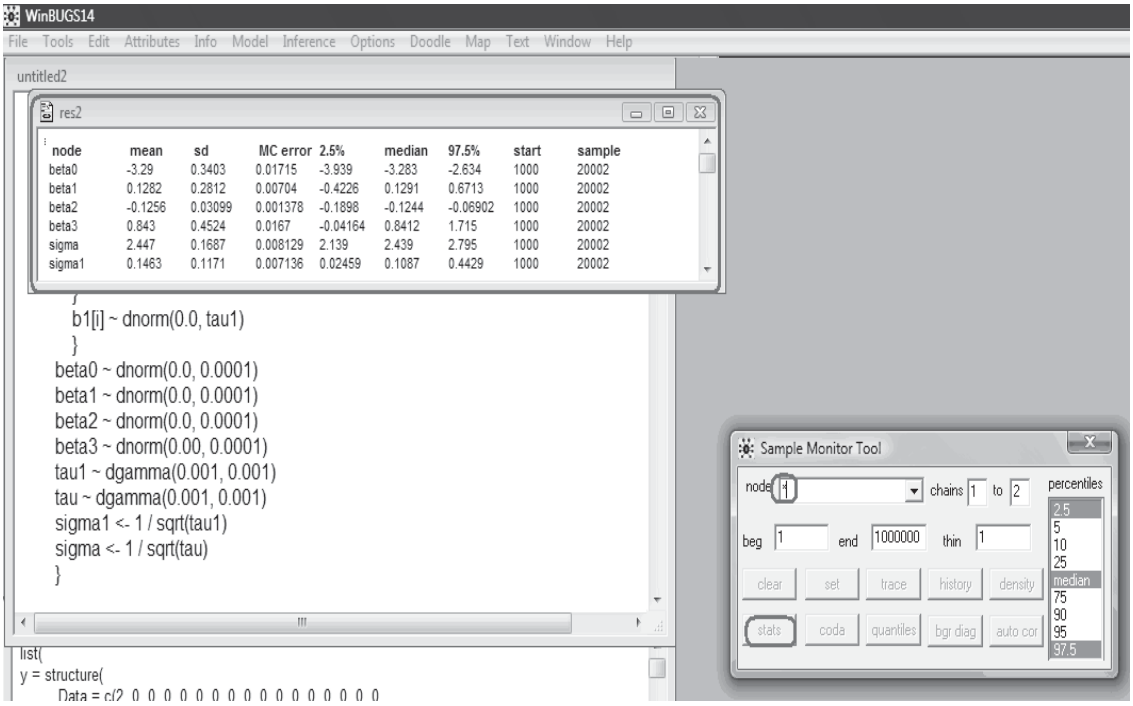

Рисунок 11 – Підсумкова статистика

Висновки

Запропоновано методику практичної побудови ієрархічних байєсівских моделей, призначених для математичного опису поведінки користувача мережі Інтернет на прикладі конкретного сайту (портал новин msnbc.com). Задачу розв'язано за допомогою програмного середовища WinBugs, яке є зручним інструментарієм для імовірнісного моделювання об'єктів у різних прикладних областях.

Результати побудови байєсівської моделі наведені в таблиці на рис. 11. Як видно з таблиці, апостеріорне математичне сподівання для стандартного відхилення мінливості в межах сесій (sigma1) близьке до нуля, що вказує, що в цілому немає істотної різнорідності між сесіями. Але ефект випадкової мінливості (sigma) має відносно велике математичне сподівання (2,447), що свідчить про значну різницю між відвідуван-

нями у випадку, якщо порівнювати їх сторінка за сторінкою. Всі параметри оцінки мають дуже невелике стандартне відхилення, що свідчить про досить сильний вплив спостережуваних ефектів.

Для створення однорідних кластерів груп користувачів найбільш важливими є ті розподіли, які передбачають тривалість візиту до головної сторінки для різних груп, та тривалість візиту на "інші сторінки". Можна передбачити, що користувачі, які відкривають сайт msnbc.com через головну сторінку, і ті, хто входить за допомогою інших сторінок, мають аналогічні апостеріорні розподіли для середньої тривалості перебування на першій сторінці та середньої тривалості перебування на інших сторінках.

Користувачі, які відвідують багато сторінок (мають велику глибину), витрачають набагато більше часу на першій сторінці, ніж на інших сторінках, незалежно від того, якими є їхні вхідні ворота для відкриття msnbc.com. Для кластеризації сесій користувачів можна скористатись критерієм – середня довжина візиту. Можна також створити тільки два кластери різних рівнів. До одного кластеру віднесемо всі сесії користувачів, які входять на msnbc.com через головну сторінку, а до іншого – всі сесії користувачів, які входять на msnbc.com через інші сторінки. Апостеріорний розподіл для середньої тривалості відвідування у цих кластерах залежить від глибини відвідування, а тому вищого рівня кластеризації можна досягти шляхом призначення одного кластера для кожного рівня глибини.

У подальших дослідженнях передбачається розширити область застосування ієрархічних байєсівских мереж і застосувати альтернативні методи формування імовірнісного висновку.

ЛІТЕРАТУРА

1. Gyftodimos E., Flach A. P. Hierarchical Bayesian Networks: a probabilistic reasoning model for structured domains / Proceedings of the ICML-2002 Workshop on Development of Representations. University of New South Wales. – Sydney, Australia, 2002. – P. 23-30.
2. Gelman, A. Bayesian Data Analysis / A. Gelman, J. B. Carlin, H. S. Stern, D. B. Rubin – Boca Raton, Florida: CHAPMAN & HALUCRC, 2004. – 668 p.

3. Gyftodimos E., Flach A. P. Hierarchical Bayesian networks: an approach to classification and learning for structured data / Department of Informatics, University of Szeged – Springer, 2004. – P. 291-300.
4. Park S. A hierarchical bayesian network for event recognition of human actions and interactions / Association For Computing Machinery Multimedia Systems Journal – The University of Texas at Austin, 2004. – P. 164-179.
5. Gyftodimos E., Flach A. P. Learning Hierarchical Bayesian Networks for human skill modelling / University of Bristol, 2004 – P. 55–62.
6. Karieauskas G. Text Categorization Using Hierarchical Bayesian Network Classifiers.– Aalborguniversity, 2002 – http://citeseer.nj.nec.com/karieauskas02text.html