УДК 373.5.07: 371.13

**Олег Вачевський,** *студент Дрогобицького державного педагогічного університету імені Івана Франка*

## РЕАЛІЗАЦІЯ АЛГОРИТМІВ ПОШУКУ НАЙКОРОТШИХ ШЛЯХІВ ТА ЇХ ПРАКТИЧНІ ВІДОМОСТІ ЗАСТОСУВАННЯ

*У статті висвітлюються оптимізаційні алгоритми застосування принципів найкоротших шляхів пошуку та їх практичного використання: з'ясовано вплив інформаційних технологій на розвиток математичного мислення студентів у процесі комп'ютерної підготовки для формування професійних компетентностей.*

***Ключові слова:*** *алгоритм, найкоротші шляхи, комп'ютерні технології, розвиток, освіта, навчання.*

***Рис. 2. Табл. 4. Літ. 7.***

**Олег Вачевский,** *студент Дрогобычского государственного педагогического университета имени Ивана Франко*

## РЕАЛИЗАЦИЯ АЛГОРИТМОВ ПОИСКА КРАТЧАЙШИХ ПУТЕЙ И ИХ ПРАКТИЧЕСКИЕ СВЕДЕНИЯ ПРИМЕНЕНИЯ

*В статье освещаются оптимизационные алгоритмы применения принципов кратчайших путей поиска и их практического использования: выяснено влияние информационных технологий на развитие математического мышления студентов в процессе компьютерной подготовки для формирования профессиональных компетенций.*

***Ключевые слова:*** *алгоритм, кратчайшие пути, компьютерные технологии, развитие, образование, обучение.*

**Oleh Vachevsky,** *Student of Drohobych State Pedagogical University by I. Franko*

## THE IMPLEMENTATION OF ALGORITHMS FOR FINDING THE SHORTEST PATHS AND THEIR PRACTICAL APPLICATION INFORMATION

*The article highlights the optimization algorithms use the principles of finding the shortest paths and their practical use: the influence of information technology on the development of mathematical thinking of students in the computer training to create professional competence.*

***Keywords:*** *algorithm, the shortest path, computer technology, development, education, training.*

**Актуальність проблеми.** Прискорення науково-технічного прогресу, зміна змісту, характеру й умов праці, автоматизація виробництва, світові досягнення науки і техніки висувають нові вимоги до всіх учасників суспільно-виробничих відносин, зазначає І .Д. Нищак [6, 4 – 5].

За таких умов важливого значення набуває розвиток комп'ютерної грамотності випускників ВНЗ, формування високого рівня технічного мислення, що відіграє важливу роль у вирішенні важливих господарських завдань та є основою розвитку виробничо-економічних процесів у суспільстві. Пріоритетний розвиток одержують електроніка, комплексна автоматизація, індустрія інформатики, технологія виробництва нових матеріалів. Ці напрями науково-технічного прогресу становлять значний вплив на ефективність засобів праці, технологічних систем у всіх галузях народного господарства. Та зростання ролі інтелектуальної власності, новаторства, що дає суттєвий економічний і значний соціальний ефект. У цьому процесі реально підтверджується одна з незаперечних переваг комп'ютерних технологій та інтелектуальної власності, великих потенціальних можливостей суспільства, зазначає М.В. Вачевський [4, 5].

**Аналіз останніх публікацій.** Проблемі пошуку алгоритмів найкоротших шляхів та розвитку комп'ютерних технологій та їх вплив на розвиток професійних компетентностей випускників ВНЗ присвячено значну кількість навчально-методичних робіт вітчизняних авторів: В.В. Анісімов, О.В. Вітюк, А.Г. Григорович, Р.С. Гуревич, О.М. Джеджула, Ю.О. Дорошенко, І .М. Лазурчак, В.В. Моштук, І .Д. Нищак, І .О. Петрицин, М.Ф. Юсупова та ін.

**Мета статті.** Дослідження і наведення

 

рекомендацій реалізації алгоритмів пошуку найкоротших шляхів та їх практичні відомості застосування в практичній діяльності, та формування в студентів теоретичних та практичних вмінь і навичок для їх використання на ринку праці.

**Виклад основного матеріалу**. Завдяки своєму широкому застосуванню, теорія про знаходження найкоротших шляхів останнім часом інтенсивно розвивається. Знаходження найкоротшого шляху – життєво необхідно і використовується практично скрізь, починаючи від знаходження оптимального маршруту між двома об'єктами на місцевості (наприклад, найкоротший шлях від будинку до університету), в системах автопілоту, для знаходження оптимального маршруту при перевезеннях, комутації інформаційного пакету в Internet і т.д. Найкоротший шлях розглядається за допомогою певного математичного об'єкту, званого графом. І снують три найбільш ефективних алгоритми знаходження найкоротшого шляху:

- алгоритм Дейкстри (використовується для знаходження оптимального маршруту між двома вершинами);

- алгоритм Форда-Бельманна (для знаходження оптимального маршруту між усіма парами вершин);

- алгоритм Флойда-Уоршалла (для знаходження найкоротшого шляху між парами вершин).

Зазначені алгоритми легко виконуються при малій кількості вершин у графі. При збільшенні їх кількості завдання пошуку найкоротшого шляху ускладнюється. Тут на допомогу приходить сучасна техніка. Комп'ютерні засоби та інформаційні технології підвищили можливості такого всеосяжного методу вивчення і створення, як моделювання об'єктів, явищ і процесів – як тих, що існують у природі, так і тих, що створюються людиною штучно. Кількість об'єктів ускладнювалися, збільшувалися, і натурне моделювання (макети споруд) стало невигідним, не економним. Тому для вивчення почали застосовувати математику. Використання математичних моделей – рівняння, нерівності, формули і тому подібне називається математичним моделюванням, для розвитку і пристосування якого потрібні були ефективні чисельні методи. Реалізувати великий потенціал математичного моделювання неможливо без потужних засобів автоматизації обчислень, якими є комп'ютери. Завдяки появі комп'ютерів і розвитку інформаційних технологій створюються методи та засоби комп'ютерного моделювання, здатні вирішувати складні практичні завдання, такі як управління великими енергетичними системами, створення достовірних прогнозів погоди або врожаю, моделювання регіональних і загальнодержавних систем, проектування літаків, кораблів тощо. Комп'ютерна модель – це розміщена в комп'ютері сукупність засобів, що реалізують концепцію обчислення. Для реалізації комп'ютерної моделі, велике значення має такий науковий напрямок, як програмування. Без нього комп'ютер це просто набір різних пристроїв, мікросхем, який не може бути корисним. Великі програми через свою складність нерідко містять помилки, які можуть стати причиною матеріальних збитків, а іноді й загрожувати життю людей (наприклад, при управлінні польотом літака). У результаті боротьби з проблемою складності програмного коду були вироблені три концепції програмування: а) об'єктно-орієнтоване програмування (ООП); б) уніфікована мова моделювання (UML); в) спеціалізовані засоби розробки програмного забезпечення. З усіх об'єктно-орієнтованих мов С++ є найбільш широко використовуваним. І саме з його допомогою в даному курсовому проекті реалізується алгоритм Дейкстри для орієнтованого графа.

**Метою даного дослідження** є програмна реалізація алгоритму пошуку найкоротшого шляху між двома будь-якими вершинами графа. Програма повинна працювати так, щоб користувач вводив кількість вершин і довжини ребер графа, а після обробки цих даних на екран виводився найкоротший шлях між двома заданими вершинами і його довжина. Необхідно передбачити різні результати пошуку, щоб програма не видавала помилок і працювала правильно. Дана програма може використовуватися в дискретній математиці для дослідження графів або в якості наочного посібника, що демонструє застосування алгоритмів на практиці.

Алгоритми пошуку найкоротших шляхів нами показано через наступні алгоритми:

**- Алгоритм Дейкстри.** Задача про найкоротший шлях полягає у знаходженні найкоротшого шляху від заданої початкової вершини до заданої кінцевої вершини. Формулювання задач про знаходження відстаней таке:

- для заданої початкової вершини *a* знайти найкоротші шляхи від *a* до всіх інших вершин.

- знайти найкоротші шляхи між усіма парами вершин.

Виявляється, що майже всі методи розв'язання задачі про найкоротший шлях від заданої початкової вершини до заданої кінцевої

вершини, також дають змогу знайти й найкоротші шляхи від вершини *a* до всіх інших вершин графа. Отже, за їх допомогою можна розв'язати задачу 1 із невеликими додатковими обчислювальними витратами. З іншого боку, задачу 2 можна розв'язати або *n* разів застосувавши алгоритм задачі 1 із різними початковими вершинами, або один раз застосувавши спеціальний алгоритм, як показано Ю.В. Нікольським [5].

**Алгоритм Дейкстри** – найефективніший алгоритм на графах відкритий нідерландським вченим Е. Дейкстром у 1959 році. Цей алгоритм знаходить найкоротшу відстань від однієї вершини графу до всіх решти. Алгоритм працює лише для графів у яких ребра не мають від'ємної ваги. Алгоритм широко застосовується в програмуванні і технологіях, наприклад, його використовують протоколи маршрутизації OSPF та IS-IS, як показано в роботі Алгоритм Дейкстри [2].

**Алгоритм Форда-Бельманна.** І сторія алгоритму зв'язана відразу з трьома незалежними математиками: Лестером Фордом, Річардом Беллманом і Едвардом Муром. Форд і Беллман опублікували алгоритм у 1956 і 1958 роках відповідно, а Мур зробив це в 1957 році. І деколи його називають алгоритмом Беллмана-Форда-Мура. Метод використовується в деяких протоколах дистанційно-векторної маршрутизації, наприклад в RIP (Routing Information Protocol – Протокол маршрутної інформації).

Так само як і алгоритм Дейкстри, алгоритм Форда-Беллмана обраховує у зваженому графі найкоротші шляхи від однієї вершини до всіх інших. Він підходить для роботи з графами, в яких є ребра від'ємної ваги. Але спектр використання алгоритму зачіпає не всі такі графи. Це пояснюється тим, що кожний наступний прохід по шляху, складеному з ребер, сума ваги яких є від'ємна, лише покращує задане значення. Безкінечне число покращень робить неможливим визначення одного конкретного значення, яке являлося б оптимальним. У зв'язку з цим алгоритм Форда-Беллмана не використовується з графами, у яких є від'ємні цикли, але він дозволяє визначити наявність таких, як зазначено в Беллмана-Форда (Електронний документ) [1].

**Алгоритм Флойда-Воршелла** – це алгоритм динамічного програмування для знаходження найкоротших відстаней між усіма вершинами зваженого орієнтованого графа. Розроблений в 1962 році Робертом Флойдом і Стівеном Воршеллом.

*Динамічне програмування* – це альтернативне вирішення задач методом "в лоб", тобто brute forc'ом або жадний алгоритм. Використовується там, де оптимальне вирішення під задачі меншого розміру може бути використано для вирішення основної задачі. В загальному виді метод виглядає так:

1. Розбір задачі на під задачі меншого розміру.
2. Знаходження оптимального вирішення під задач рекурсивно.
3. Використання результатів під задач для вирішення основної задачі.

Для знаходження найкоротших шляхів між всіма вершинами графа використовується не перебір всіх можливостей, що приведе до використання часу та затрат більшого об'єму пам'яті, а динамічне програмування, тобто під задачі, які будуть потрібні для вирішення головної задачі, прораховуються завчасно і потім використовуються [1].

Практичні відомості та застосування алгоритмів пошуку, нами приведено на відповідних графах алгоритмів.

**Алгоритм Дейкстри.** Розглянемо деякий орієнтований граф, для якого потрібно знайти найкоротший маршрут від вершини 1 до всіх інших вершин. Для розв'язку задач такого типу доцільно використовувати алгоритм Дейкстри.

У наступному алгоритму, початковій вершині (вершина під номером 1) присвоюємо постійну мітку [0 ; -] після чого переходимо до першого етапу.

*Eman 1*: з вершини 1 існує орієнтоване ребро до вершин 2, 4 і 5 (Рис. 1). Обчислимо для даних вершин відповідні мітки. В результаті отримаємо наступну таблицю (Табл. 1):

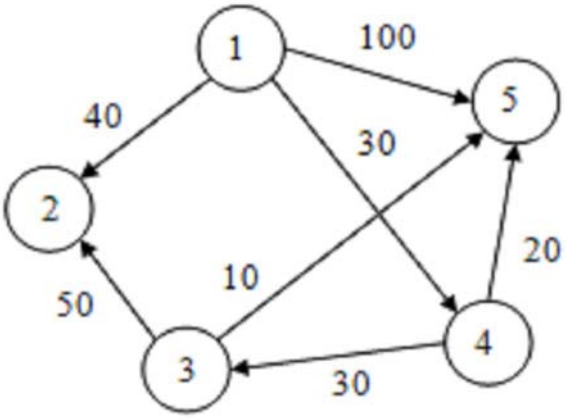


**Рис. 1. Орієнтований граф, для якого потрібно знайти найкоротший маршрут**

Серед вершин з тимчасовими мітками, виберемо ту, значення відстані для якої є найменшим. В нашому випадку такою є вершина 4, з відстанню $u_4 = 30$ . Міняємо статус даної вершини на "постійна" і переходимо до другого етапу.

**Таблиця 1.**

| Номер вершини | Мітка | Статус мітки |
|---|---|---|
| 1 | [0, -] | Постійна |
| 2 | [0 + 40,1]=[40,1] | Тимчасова |
| 4 | [0 + 30,1]=[30,1] | Тимчасова |
| 5 | [0 + 100,1]=[100,1] | Тимчасова |

*Eman 2*: з четвертої вершини існує орієнтоване ребро до вершин 3 і 5. Обчислюємо для них мітки, після чого таблиця набуде наступного вигляду (Табл. 2.):

На четвертому етапі, ми отримали таблицю, в якій з тимчасовою міткою залишилась тільки вершина 3. І виходячи з того, що з даної вершини можна потрапити у вершину 2 або 5, статус яких

**Таблиця 2.**

| Номер вершини | Мітка | Статус мітки |
|---|---|---|
| 1 | [0, -] | Постійна |
| 2 | [0 + 40,1]=[40,1] | Тимчасова |
| 4 | [0 + 30,1]=[30,1] | Постійна |
| 5 | [30 + 20,4]=[50,4] | Тимчасова |
| 3 | [30 + 30,4]=[60,4] | Тимчасова |

В даній таблиці, тимчасову мітку [100, 1] для вершини 5, отриману на попередньому етапі, замінено на нову [50, 4], також тимчасову мітку. Це говорить про те, що до вершини 5 знайдено більш короткий маршрут, який проходить через вершину 4. Після чого, аналогічно першому етапу, вибираємо вершину значення відстані для якої є найменшим (вершина під номером 2) і міняємо її статус на постійну.

*Eman 3*: від вершини 2 не можливо потрапити до інших вершин графа, тому на даному етапі ми отримаємо аналогічну другому етапу таблицю, тільки статус вершини 2 змінено на постійна (Табл. 3):

становить – постійна, то на цьому процес обчислень за алгоритмом Дейкстри закінчується.

Таким чином ми отримали маршрут найкоротшої довжини, який починається у вершині 1 і обходить всі вершини заданого графа (Рис. 2.) [4].

Якщо граф подано матрицею суміжності, складність алгоритму Дейкстри становить $O(n^2)$. Коли кількість дуг значно менша ніж $n^2$, то найкраще подавати орієнтований граф списками суміжності. Тоді алгоритм можна реалізувати зі складністю O(m lgn), що в цьому разі істотно менше ніж $O(n^2)$.

**Таблиця 3.**

| Номер вершини | Мітка | Статус мітки |
|---|---|---|
| 1 | [0, -] | Постійна |
| 2 | [0 + 40,1]=[40,1] | Постійна |
| 4 | [0 + 30,1]=[30,1] | Постійна |
| 5 | [30 + 20,4]=[50,4] | Тимчасова |
| 3 | [30 + 30,4]=[60,4] | Тимчасова |

Серед вершин, статус мітки яких не дорівнює постійна, вибираємо ту, для якої значення відстані $u_i$ (i=1,5) є найменшим. Тобто вершину 5, для якої $u_5$=50. Міняємо її статус на постійну і переходимо до четвертого етапу.

*Eman 4*: з вершини 5 не існує орієнтованих ребер до інших вершин графа, тому таблиця міток залишається незмінною, тільки статус вершини 5 замінено на постійну (Табл. 4).

**Алгоритм Форда-Бельманна.** Заданий орієнтований зважений граф, що містить, можливо, дуги негативного ваги. Необхідно знайти ваги найкоротших шляхів від вершини v до всіх інших вершин.

Нехай $N$ – кількість вершин графа, $M$ – кількість дуг, $L_{i,j}$ – вага дуг , які виходять з вершини $i$ у вершину $j$. Позначимо за $T_{i,j}$ вагу найкоротшого шляху із $v$ у $j$, який містить не

**Таблиця 4.**

| Номер вершини | Мітка | Статус мітки |
|---|---|---|
| 1 | [0, -] | Постійна |
| 2 | [0 + 40,1]=[40,1] | Постійна |
| 4 | [0 + 30,1]=[30,1] | Постійна |
| 5 | [30 + 20,4]=[50,4] | Постійна |
| 3 | [30 + 30,4]=[60,4] | Тимчасова |

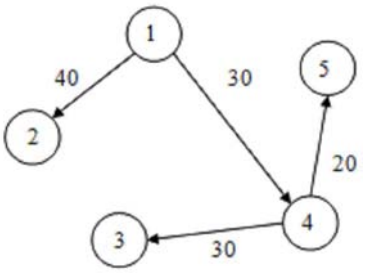


**Рис. 2. Найкоротший маршрут**

більше *і дуг. У випадку відсутності такого шляху положимо* $T_{i,j} = \infty$ , де безкінечність, це деяке значення, яке є більшим від усіх можливих відстаней. При $i = 0$ існує єдиний допустимий шлях – шлях з вершини *v в неї саму, який має нульову вагу. Відповідно, при* $j` \neq v$ $T_{j,0} = \infty$ . Нехай тепер $i > 0$. Зрозуміло, що найкоротший шлях з вершини *v* у вершину *j складається з найкоротшого шляху із v* у деяку вершину *p* та дуги *(p, j).*

Тепер можна записати наступне рекурентне співвідношення (1):

$$T_{I,J} = \begin{cases} 0, \ if \ i = 0, j = v; \\ \infty, if \ i = 0, j \neq v; \\ \min(\infty \min k \in V, (k, j) \in E, T_{i-1,k} < \infty (T_{i-1,k} + l_{k,j})), if \ i > 0 \end{cases} \quad (1)$$

Тут *V* – множина вершин графа, *E* – множина дуг.

Відмітимо, що оптимальний шлях не повинен містити циклів. На справді, якщо у нас є деякі шляхи з циклом, то, викинувши цикл, ми дістанемо шлях меншої або рівної ваги. Тоді максимальна кількість дуг у шляху буде рівна *N-1* (у протилежному випадку у шляху появиться цикл.)

Припустимо, що нам потрібно ввести найкоротший шлях. Тоді для кожної під задачі необхідно буде зберігати передостанню вершину шляху, тобто значення *k* (див. Рекурентне співвідношення). Це означає, що буде потрібна матриця розмірності *NM* [7].

Час роботи алгоритму Белмана-Форда становить O(nm), де n – кількість вершин та m – кількість ребер графу G. Це більше, ніж для алгоритму Дейкстри, для якого час роботи визначається як O(m lgn).

Отже, якщо всі цикли в графі мають не протилежну вагу, то для знаходження ваги найкоротших шляхів буде достатньо *N-1* ітерацій. Але в протилежному випадку ситуація зміниться. Якщо у шляху є цикл протилежної ваги, то ми можемо рухатися по ньому безкінечно довго, кожен раз зменшуючи довжину шляху. Провівши тестову *N-у* ітерацію ми побачимо, що якщо в шляхах є цикл з протилежною вагою, то відстань шляху зменшиться відповідно до попередніх результатів.

**Алгоритм Флойда-Уоршалла.**

1. Присвоювання початкових значень. Пронумерувати вершини графа G цілими числами від 1 до n. Побудувати матрицю $W_{(0)}$, задавши кожний її (I, j)-й елемент таким, що дорівнює вазі дуги, котра з'єднує вершину I з вершиною j. Якщо в графі G ці вершини не з'єднано дугою, то виконати $W_y^{(0)} = \infty$ . Крім того, для всіх I виконати $W_u^{(0)} = 0$.

2. Цикл по всіх k, що послідовно набуває значення 1, 2, …, n, визначити за елементами матриці $W_{(k-1)}$ елементи матриці $W_{(k)}$, використовуючи рекурентне співвідношення (2):

$$L_{i,v} = \min \left\{ \begin{array}{c} L_{(i-1),v} \\ \min\limits_{(u,v) \in E} \{ L_{(i-1),u} + w(u,v) \} \end{array} \right\} \quad (2)$$

Якщо під час роботи алгоритму для якихось k та I виявиться, що $w^{(k)}_{ii} < 0$, то в графі є цикл із від'ємною довжиною який містить вершину I. Тоді роботу алгоритму потрібно припинити.

Якщо заздалегідь відомо, що в графі немає циклів із від'ємною довжиною, то обсяг обчислень можна дещо зменшити. У цьому разі для всіх I та всіх k має бути $w^{(k)}_{ii} = 0$, тому не потрібно обчислювати діагональні елементи матриць $W_{(1)}$, $W_{(2)}$, … $W_{(n)}$.

Отже, обчислюючи матрицю, немає потреби ондарівкийати елементи k-го рядка й k-го стовпця матриці $W_{(k-1)}$. Отже, у матриці $W_{(k)}$ за формулою (3) потрібно обчислювати лише $n^2 - 3n+2$ елементів. Очевидно, що складність алгоритму Флойда-Уоршала становить $O(n^3)$ [3].

$$\omega_{ij}^{k} = \min\left\{\omega_{ik}^{(k-1)} + \omega_{kj}^{(k-1)}, \omega_{ij}^{(k-1)}\right\} \quad (3)$$

**Висновки.** Алгоритми пошуку найкоротших шляхів широко застосовуються і розвиваються в наш час. Кожен з нас завжди шукає найкоротший шлях для вирішення певної задачі. Правильне використання алгоритмів дає змогу знаходити найкоротший шлях з найменшими затратами часу та ресурсів. В даному курсовому проекті були розглянуті три алгоритми пошуку найкоротших шляхів у графах:

- алгоритм Дейкстри (використовується для знаходження оптимального маршруту між двома вершинами);
- алгоритм Форда-Бельманна (для знаходження оптимального маршруту між усіма парами вершин);
- алгоритм Флойда-Уоршалла (для знаходження найкоротшого шляху між парами вершин).

До цих алгоритмів були створенні програми, які можуть використовуватися у вивчені курсу дискретної математики, як наочний приклад роботи алгоритму. Крім цього, використовуючи засоби алгоритмів пошуку найкоротших шляхів програмування, їх можна застосовувати для подальшої розробки автоматизованих навчальних систем.

*1. Алгоритм Беллмана-Форда [Електронний документ]. – Режим доступу: URL: http://uk.wikipedia.org/wiki/Алгоритм_Беллмана-Форда.*

*2. Алгоритм Дейкстри [Електронний документ]. – Режим доступу: URL: http://uk.wikipedia.org/wiki/Алгоритм_Дейкстри*

*3. Бондарів В.М. Програмування на С + +. – Х: "Компанія СМІТ", 2004*

*4. Вачевський М.В. Соціально-економічні аспекти використання інтелектуальної власності в сучасних умовах. Навчальний посібник. // М. Вачевський. – К.: ЦИКЛ, 2004. – 376 с.*

*5. Ю.В. Нікольський. Дискретна математика. Київ. Видавнича група BHV. 2007 р.*

*6. Нищак І.Д. Інформаційні технології як засіб розвитку технічного мислення майбутніх учителів трудового навчання у процесі графічної підготовки. Монографія. // І. Нищак. – Дрогобич, ДДПУ, 2011. – 158 с.*

*7. Приклад знаходження дерева мінімальної вартості для орієнтованого графа за алгоритмом Дейкстри [Електронний документ]. – Режим доступу: URL: http://www.mathros.net.ua/?p=1472 – пояснення алгоритму Дейкстри.*

Стаття надійшла до редакції 10.01.2014

*"Любов – це не просто солоденьке слово, але праця і сльози. В світі не вистачає 3-х мільйонів лікарів, то ж станьте лікарями. Більше мільярда людей – не освічені, то ж станьте вчителями.*

*Станьте працівниками в будь-якій галузі. Протестуйте, але для того, щоб творити. Творіть щастя інших людей.*

*Правдива любов бачить відродження Христового лика на обличчі страждучого, бідного, гонимого. Дуже довго люди жили один біля другого, не цікавлячись одне одним. Тепер ми маємо навчитися жити один для другого. Адже єдина істина – любити один одного".*

*Рауль Фольро*
*християнський письменник "Ліги боротьби з хворобою, голодом та бідністю", – 30 разів об'їхав цілий світ, закликаючи різні народи об'єднатися в ім'я милосердя та співчуття.*