Pancake Sort

1. Introducere

Problema sortarii Pancake, in ciuda numelui sau aparent ludic, reprezinta o provocare algoritmica si matematica semnificativa. In esenta, aceasta descrie scenariul sortarii unei stive de clatite de diferite dimensiuni, asezate initial intr-o ordine aleatorie, cu scopul de a le aranja de la cea mai mica la cea mai mare (sau invers), cea mai mica fiind deasupra. Unicitatea problemei consta in singura operatie permisa: introducerea unei spatule oriunde in stiva si rasturnarea tuturor clatitelor situate deasupra spatulei.[1] Aceasta operatie este cunoscuta formal sub denumirea de "inversare de prefix". Spre deosebire de majoritatea algoritmilor de sortare traditionali, care se concentreaza pe minimizarea numarului de comparatii intre elemente, obiectivul principal in sortarea Pancake este minimizarea numarului total de rasturnari.[1] Desi poate fi perceputa initial ca un simplu joc sau un instrument educational, sortarea Pancake poseda conexiuni profunde si surprinzatoare cu domenii stiintifice si tehnologice avansate, inclusiv biologia computationala si arhitectura retelelor de procesoare paralele.[2]
Importanta sa ca problema fundamentala in stiinta calculatoarelor deriva din intrebarile pe care le ridica referitor la eficienta algoritmilor care opereaza sub constrangeri severe si la limitele computationale inerente unor astfel de probleme.[4] Caracterul "jucaus" al descrierii initiale, cu clatite si o spatula, face ca problema sa fie extrem de accesibila si memorabila. Aceasta accesibilitate serveste drept o poarta de intrare excelenta catre concepte algoritmice mult mai profunde. Desi analogia cu clatitele o face usor de inteles la un nivel superficial [1], intrebarea fundamentala – cum se poate sorta stiva folosind un numar minim de rasturnari – se traduce intr-o problema matematica si algoritmica non-triviala.[2] Aceasta tranzitie de la o descriere simpla la o problema complexa demonstreaza modul in care conceptele fundamentale din stiinta calculatoarelor pot fi ilustrate prin exemple aparent simple, dar cu un substrat teoretic bogat. Mai mult, contrastul fundamental intre obiectivul sortarilor clasice (minimizarea comparatiilor) si cel al sortarii Pancake (minimizarea operatiilor de rasturnare) defineste un spatiu distinct de probleme. Algoritmii de sortare standard, precum MergeSort sau QuickSort, sunt adesea evaluati prin prisma numarului de comparatii efectuate.[2] In schimb, sortarea Pancake ignora costul comparatiilor si se concentreaza exclusiv pe numarul de inversari de prefix.[2] Aceasta schimbare de paradigma forteaza gandirea algoritmica sa se adapteze la un set diferit de costuri operationale si constrangeri, necesitand dezvoltarea unor strategii si tehnici de analiza specifice.

2. Definitia si Mecanismul Sortarii Pancake
2.1. Ce este Sortarea Pancake?

Din punct de vedere formal, sortarea Pancake este problema matematica si algoritmica a aranjarii unei secvente dezordonate de elemente distincte (reprezentate ca o stiva de clatite

sau un sir de numere) intr-o ordine specifica, de obicei crescatoare, utilizand exclusiv operatia de "flip" sau "inversare de prefix".[1] Un "flip(k)", unde k este un indice (sau numarul de elemente de la varf), inverseaza ordinea primelor k elemente ale secventei. De exemplu, daca avem stiva [c1, c2, c3, c4, c5] si aplicam flip(3), stiva devine [c3, c2, c1, c4, c5].
Obiectivul principal nu este de a sorta elementele efectuand cat mai putine comparatii, ci de a realiza sortarea intr-un numar cat mai mic de operatii de rasturnare.[1] Numarul minim de rasturnari necesar pentru a sorta cea mai defavorabila configuratie initiala de n clatite este o valoare importanta in teoria acestei probleme si este denumit "numarul Pancake", notat cu P(n).

2.2. Cum Functioneaza? Un Algoritm Standard

Cel mai frecvent prezentat algoritm pentru sortarea Pancake, si cel mai intuitiv, este similar ca strategie cu algoritmul Selection Sort si functioneaza iterativ.[1] Pentru o stiva (sau un subsir) de curr_size clatite nesortate, algoritmul parcurge urmatorii pasi:

1. **Gasirea elementului maxim:** Se identifica elementul cu cea mai mare valoare (cea mai mare clatita) in portiunea nesortata curenta, adica printre primele curr_size elemente. Fie mi indexul acestui element (considerand indexarea de la 0).
2. **Aducerea maximului la pozitia corecta:** Daca elementul maxim nu se afla deja la pozitia sa finala in cadrul portiunii curente (adica la indexul curr_size - 1): a. **Aducerea maximului la varf:** Daca elementul maxim nu este deja la varful stivei (la indexul 0), se efectueaza o rasturnare flip(mi + 1) (deoarece flip opereaza cu numarul de elemente, nu cu indexul 0-based). Aceasta operatie aduce elementul maxim in varful stivei.[1] b. **Mutarea maximului la baza portiunii curente:** Se efectueaza o rasturnare flip(curr_size). Aceasta operatie muta elementul maxim (care se afla acum la varf) la baza portiunii curente de curr_size elemente, adica la pozitia sa corecta sortata relativ la aceasta portiune.[1]
3. **Reducerea problemei:** Se considera ca elementul maxim plasat corect este acum sortat. Dimensiunea portiunii nesortate, curr_size, se reduce cu 1, iar procesul se repeta pentru restul stivei pana cand intreaga stiva este sortata (curr_size devine 1).

Performanta acestui algoritm standard este bine definita. El garanteaza sortarea stivei efectuand cel mult 2n−3 rasturnari in cel mai rau caz, unde n este numarul total de clatite.[2] Din punct de vedere al complexitatii temporale, algoritmul este O(n2). Aceasta se datoreaza faptului ca pentru fiecare din cei n pasi principali (plasarea corecta a unui element), este necesara o cautare a elementului maxim, care dureaza O(n) in portiunea nesortata, si cel mult doua operatii de rasturnare, fiecare putand dura O(n) pentru a inversa elementele. Complexitatea spatiala este O(1), deoarece algoritmul opereaza pe loc, fara a necesita structuri de date auxiliare semnificative.
Pentru a ilustra functionarea, sa consideram sirul arr = [7]:

- Initial: ``, curr_size = 4. Maximul in arr[0...3] este 4, la indexul 2.
  - Aducem 4 la varf: flip(2+1) (k=3) pe =>.
  - Mutam 4 la pozitia curr_size-1 (index 3): flip(4) pe =>. Acum 4 este la locul lui.

- curr_size = 3. Consideram ``. Maximul este 3, la indexul 1.
  - Aducem 3 la varf: flip(1+1) (k=2) pe =>.
  - Mutam 3 la pozitia curr_size-1 (index 2): flip(3) pe =>. Acum 3 este la locul lui.
- curr_size = 2. Consideram ``. Maximul este 2, la indexul 0.
  - 2 este deja la varf. Nu este necesar primul flip.
  - Mutam 2 la pozitia curr_size-1 (index 1): flip(2) pe =>. Acum 2 este la locul lui.
- curr_size = 1. Consideram ``. Este sortat.

Desi acest algoritm este intuitiv si relativ usor de implementat , el nu garanteaza gasirea solutiei cu numarul minim absolut de rasturnari. Strategia sa este una "greedy" (lacoma): la fiecare pas, se concentreaza pe plasarea celui mai mare element nesortat la locul lui, fara a lua in considerare impactul pe termen lung al rasturnarilor asupra configuratiei generale.[1]
Faptul ca problema gasirii numarului minim de rasturnari este NP-hard [4] inseamna ca este putin probabil sa existe un algoritm eficient care sa gaseasca intotdeauna solutia optima. Astfel, algoritmul standard, cu cele 2n−3 rasturnari, serveste drept o euristica – ofera o solutie corecta intr-un numar rezonabil de pasi, dar nu neaparat cea mai buna posibila. Acest decalaj intre o solutie simpla, polinomiala, si solutia optima, dificil de gasit, este o caracteristica comuna multor probleme computationale complexe.
Operatia de "flip" in sine este puternica, deoarece poate modifica pozitiile relative ale unui numar mare de elemente simultan, inversand un intreg prefix al stivei. Spre deosebire de o operatie simpla de "swap" (schimbare a doua elemente), care are un impact localizat, un "flip" are consecinte mai ample asupra aranjamentului. Aceasta putere vine insa cu costul unei predictibilitati reduse a efectelor pe termen lung ale unei secvente de rasturnari. Fiecare flip poate desface partial progresul realizat de operatiile anterioare sau poate crea configuratii intermediare a caror evaluare este dificila in contextul gasirii drumului optim catre starea sortata. Aceasta natura neliniara si cu impact extins a operatiei fundamentale contribuie semnificativ la complexitatea intrinseca a problemei sortarii Pancake.

3. Istoric si Contributii Notabile

Originea problemei:
Problema sortarii Pancake a fost adusa in atentia comunitatii matematice si popularizata pentru prima data in 1975 de catre Jacob E. Goodman. Acesta a publicat o nota in jurnalul American Mathematical Monthly sub pseudonimul "Harry Dweighter".6 In nota sa, Goodman a descris intr-un mod pitoresc scenariul unui chelner dintr-un restaurant aglomerat care incearca sa sorteze o stiva de clatite de diferite marimi, avand la dispozitie o singura mana libera si putand doar sa ridice o portiune superioara a stivei, sa o inverseze si sa o puna la loc.6 Aceasta formulare accesibila a fost esentiala pentru raspandirea problemei dincolo de cercurile strict matematice. Abordarea narativa, ancorata intr-un context cotidian, face problemele abstracte mai atractive si mai usor de diseminat, contribuind la longevitatea si interesul continuu pentru ele.
Contributiile lui Bill Gates si Christos Papadimitriou:
Un moment important in istoria problemei a fost publicarea, in 1979, a lucrarii "Bounds for Sorting by Prefix Reversal" de catre William H. (Bill) Gates si Christos H. Papadimitriou.4

Aceasta lucrare este notabila nu doar pentru contributiile sale stiintifice, ci si pentru faptul ca este singura lucrare stiintifica publicata de Bill Gates, viitorul fondator al Microsoft.11 Gates si Papadimitriou au stabilit primele limite matematice semnificative pentru numarul Pancake P(n). Ei au demonstrat ca $P(n) \le (5n+5)/3$ (o limita superioara, adica un numar de rasturnari care este intotdeauna suficient) si ca $P(n) \ge 17n/16$ pentru valori ale lui n care sunt multiplu de 16 (o limita inferioara, adica un numar de rasturnari care este uneori necesar).2 De asemenea, in aceeasi lucrare, Gates si Papadimitriou au introdus o varianta importanta a problemei, cunoscuta sub numele de "Problema Clatitei Arse" (Burnt Pancake Problem).6
Implicarea lui Bill Gates, o figura proeminenta in industria software, chiar si in stadiile incipiente ale carierei sale, intr-o problema de matematica teoretica subliniaza intersectia fertila dintre gandirea matematica abstracta si provocarile practice din domeniul calculatoarelor. Faptul ca un viitor lider al revolutiei software si-a dedicat timpul studiului sortarii clatitelor este un detaliu biografic mai putin cunoscut, dar care adauga o nota de curiozitate si prestigiu problemei.[6] Acest episod demonstreaza cum problemele matematice fundamentale pot atrage si forma minti care ulterior au un impact major in domenii aplicate ale tehnologiei. Exista chiar si relatari despre un concurs informal la Universitatea Harvard, legat de gasirea celui mai eficient algoritm pentru aceasta problema, la care Gates ar fi participat si ar fi stabilit un record care a rezistat decenii, adaugand un element de "folclor academic" istoriei sortarii Pancake.[11]

4. Probleme Abordate si Complexitate

4.1. Problema Matematica Fundamentala

Dincolo de algoritmul specific de sortare, problema Pancake ridica intrebari matematice fundamentale. Problema centrala este determinarea valorii exacte a lui P(n), numarul minim de rasturnari necesare pentru a sorta cea mai "dificila" permutare initiala de n elemente.[2] Aceasta intrebare este echivalenta cu gasirea diametrului grafului Pancake, un graf in care nodurile reprezinta toate permutarile posibile ale celor n elemente, iar o muchie exista intre doua permutari daca una poate fi transformata in cealalta printr-o singura operatie de rasturnare de prefix.[2] Cercetarea se concentreaza atat pe dezvoltarea de algoritmi care pot sorta o stiva data folosind un numar cat mai mic de rasturnari, cat si pe stabilirea unor limite teoretice (inferioare si superioare) cat mai stranse pentru P(n).

4.2. Complexitatea Algoritmica

- Numarul Pancake P(n):
  Cercetarile intensive de-a lungul deceniilor au condus la stabilirea unor limite pentru P(n). Se stie ca P(n) se situeaza intre $15n/14$ (limita inferioara demonstrata de Heydari si Sudborough) si $18n/11$ (limita superioara actuala, obtinuta de o echipa de la Universitatea din Texas la Dallas, care a imbunatatit limita initiala de $(5n+5)/3$ propusa de Gates si Papadimitriou).2 Cu toate acestea, determinarea valorii exacte a lui P(n) pentru un n arbitrar ramane o problema deschisa in matematica si stiinta

calculatoarelor.4 Decalajul persistent intre aceste limite, desi s-a redus considerabil de la primele estimari, sugereaza ca tehnicile actuale de analiza ar putea sa-si fi atins o anumita limita si ca noi abordari matematice ar putea fi necesare pentru a inchide complet acest interval si a gasi o formula exacta pentru P(n).

- NP-Hardness:
  O alta dimensiune a complexitatii problemei se refera la dificultatea de a gasi solutia optima pentru o instanta data. S-a demonstrat ca problema gasirii secventei minime de rasturnari necesare pentru a sorta o stiva particulara de clatite este NP-hard.4 Aceasta clasificare are implicatii profunde: inseamna ca, pentru valori mari ale lui n, este foarte improbabil sa existe un algoritm care sa poata gasi intotdeauna solutia optima intr-un timp de executie polinomial in raport cu n (presupunand ca P!= NP). Aceasta NP-hardness justifica utilizarea unor algoritmi euristici, precum cel descris anterior (care garanteaza sortarea in cel mult 2n−3 rasturnari si ruleaza in timp O(n2)), chiar daca acestia nu ofera intotdeauna numarul minim absolut de rasturnari.2 Pentru instante mari ale problemei, cercetatorii trebuie adesea sa se multumeasca cu algoritmi aproximativi care ofera solutii "suficient de bune" intr-un timp rezonabil, sau sa se concentreze pe rezolvarea optima a unor cazuri particulare sau a unor instante de dimensiuni mici.
- Valori cunoscute ale P(n) pentru n mic:
  Pentru a oferi o perspectiva concreta asupra complexitatii problemei, iata un tabel cu valorile exacte cunoscute ale lui P(n) pentru valori mici ale lui n, conform datelor disponibile pana in 2016 8:

n	**P(n)**
1	0
2	1
3	3
4	4
5	5
6	7
7	8
8	9
9	10
10	11
11	13
12	14
13	15
14	16
15	17
16	18
17	19
18	20
19	22

Acest tabel ancoreaza discutia teoretica despre limitele asimptotice in exemple concrete si ilustreaza cresterea non-liniara a lui P\(n\). De asemenea, faptul ca P\(20\) era necunoscut la momentul referintei [8] subliniaza dificultatea calcularii acestor valori chiar si pentru `n` relativ mic, intarind ideea ca problema generala este deschisa si complexa.

5. Relevanta Sortarii Pancake

Relevanta problemei sortarii Pancake depaseste curiozitatea matematica, extinzandu-se in diverse domenii ale stiintei calculatoarelor teoretice si aplicate.

5.1. In Stiinta Calculatoarelor Teoretica

Problema serveste ca un excelent studiu de caz pentru designul si analiza algoritmilor, precum si pentru teoria complexitatii. Ea ilustreaza provocari legate de optimizarea sub constrangeri severe (doar operatia de flip este permisa), analiza limitelor inferioare si superioare pentru performanta algoritmilor si intelegerea claselor de complexitate precum NP-hard.[3] Studiul sortarii Pancake stimuleaza dezvoltarea de noi tehnici algoritmice si matematice pentru analiza permutarilor si a proprietatilor grafurilor asociate acestora.

5.2. Aplicatii Practice

- **Retele de procesare paralela:** Operatia de inversare de prefix si problema sortarii Pancake au legaturi directe cu algoritmii de rutare a datelor in anumite tipuri de retele de interconectare pentru procesoare paralele, cunoscute sub numele de retele Pancake (Pancake networks).[2] Graful Pancake, care modeleaza aceste retele, poseda proprietati topologice atractive, cum ar fi un grad mic al nodurilor (numarul de conexiuni directe ale unui procesor) si un diametru relativ mic (distanta maxima intre oricare doua procesoare), facandu-l un candidat interesant pentru arhitecturi de calcul paralel.[2] Un algoritm de sortare prin inversari de prefix poate servi, in acest context, ca un algoritm eficient de rutare a mesajelor in retea.[6] Aceasta conexiune sugereaza ca principiile derivate din studiul sortarii Pancake pot influenta designul arhitecturilor de calcul de inalta performanta, unde eficienta comunicarii si rearanjarii datelor intre procesoare este critica.
- Biologie computationala (Rearanjamente Genomice):
  Una dintre cele mai surprinzatoare si fructuoase aplicatii ale sortarii Pancake se gaseste in biologia computationala, in special in studiul rearanjamentelor genomice.3 Exista o analogie puternica intre sortarea unei stive de clatite si problema sortarii prin inversiuni

a segmentelor de gene de-a lungul unui cromozom. Genele pe un cromozom pot fi reprezentate ca o secventa ordonata, iar evenimentele evolutionare pot include inversiuni ale unor segmente de ADN, schimband ordinea genelor. Numarul minim de astfel de inversiuni necesar pentru a transforma secventa genetica a unei specii in cea a altei specii poate fi folosit ca o masura a distantei evolutionare dintre ele.12 Aceasta aplicabilitate este un exemplu remarcabil al modului in care problemele matematice abstracte pot oferi modele puternice pentru intelegerea proceselor biologice complexe, cum ar fi evolutia genomului, demonstrand universalitatea anumitor structuri si operatii matematice.

5.3. Variante Importante ale Problemei

- Problema Clatitei Arse (Burnt Pancake Problem):
  Aceasta varianta, introdusa de Gates si Papadimitriou 6, adauga un nivel suplimentar de complexitate. Fiecare clatita are o fata "arsa" si una "ne-arsa" (sau "aurie"). Scopul este de a sorta clatitele nu doar dupa marime, ci si astfel incat toate sa ajunga cu fata ne-arsa in sus (sau, echivalent, cu fata arsa in jos).2 O operatie de flip in acest context nu doar inverseaza ordinea clatitelor din prefixul selectat, ci si "intoarce pe dos" fiecare clatita din acel prefix, schimband starea fetei sale superioare (din arsa in ne-arsa si invers).5
  Pentru a modela aceasta situatie, se folosesc permutari semnate, unde un element i reprezinta o clatita cu fata ne-arsa in sus, iar un element negativ i' (sau -i) indica faptul ca clatita i este cu fata arsa in sus.2 Problema Clatitei Arse este si mai relevanta pentru analogia cu rearanjamentele genomice, deoarece genele pe un cromozom au si o orientare (directia de citire 5'-3' sau 3'-5').5
  Si pentru aceasta problema au fost studiate limite. De exemplu, Cohen si Blum au stabilit limite inferioare de $3n/2$ si superioare de $2n-2$ pentru numarul de rasturnari necesare in cel mai rau caz.6 Complexitatea transformarii unui sir semnat compatibil intr-un altul folosind un numar minim de inversari de prefix semnate (problema clatitei arse pe siruri) a fost demonstrata ca fiind NP-completa de catre Chitturi in 2011.2
  Un experiment deosebit de interesant legat de Problema Clatitei Arse este construirea unui "calculator bacterian". In 2008, un grup de studenti a programat bacterii E. coli pentru a rezolva o instanta simpla a acestei probleme. Bacteriile au fost modificate genetic pentru a efectua inversiuni ale unor segmente de ADN, analoge rasturnarilor de clatite arse, raportand solutia prin dezvoltarea rezistentei la antibiotice.2 Acest experiment, desi la o scara mica, deschide perspective fascinante asupra calculului biologic si a utilizarii sistemelor vii pentru a rezolva probleme combinatorice, exploatand paralelismul masiv inerent replicarii celulare si potentialul pentru costuri reduse in rezolvarea anumitor tipuri de probleme.12
- Problema Deterministica (Topswaps - Conway):
  O alta varianta, mai putin discutata dar distincta, este cea propusa de John Conway, cunoscuta si sub numele de "topswaps". In aceasta versiune, numarul de elemente din prefixul care este inversat este intotdeauna egal cu valoarea numerica a primului

element (clatita de la varf). Procesul se opreste atunci cand elementul cu valoarea 1 ajunge la varful stivei.6

6. Concluzii

Sortarea Pancake, desi porneste de la o analogie simpla si accesibila, se dovedeste a fi o problema matematica si algoritmica profunda, cu o bogata istorie si cu implicatii semnificative in diverse ramuri ale stiintei calculatoarelor. Ea serveste nu doar ca un instrument educational valoros pentru ilustrarea unor concepte precum operatiile restrictionate, analiza complexitatii si NP-hardness-ul, ci si ca o sursa continua de probleme de cercetare interesante si provocatoare.

Importanta sa teoretica este dublata de aplicatii practice surprinzatoare, in special in modelarea rearanjamentelor genomice din biologia computationala si in designul retelelor de interconectare pentru procesoare paralele. Aceste conexiuni demonstreaza puterea modelelor matematice abstracte de a oferi perspective asupra unor sisteme complexe din lumea reala.

Cu toate acestea, multe aspecte ale sortarii Pancake raman probleme deschise. Determinarea unei formule exacte pentru numarul Pancake P(n) pentru un n general continua sa eludeze matematicienii si informaticienii.[4] De asemenea, desi se cunosc algoritmi care ofera solutii aproximative, gasirea unor algoritmi de aproximare cu garantii de performanta mai bune pentru numarul minim de rasturnari este un domeniu activ de cercetare. Variantele problemei, cum ar fi Problema Clatitei Arse, prezinta propriile lor provocari, complexitatea exacta a sortarii unei permutari de clatite arse (nu siruri) fiind inca necunoscuta.[5]

Longevitatea problemei sortarii Pancake ca subiect de cercetare, de la formularea sa initiala in 1975 si pana in prezent, cu imbunatatiri si noi descoperiri care apar periodic [2], demonstreaza ca problemele fundamentale, chiar si cele cu enunturi simple, pot ascunde o bogatie si o complexitate care sustin interesul stiintific pe termen lung. Studiul sortarii Pancake si al variantelor sale poate, de asemenea, sa inspire abordari noi pentru alte probleme de sortare sau rearanjare cu operatii restrictionate, care pot aparea in diverse domenii ale stiintei si ingineriei, demonstrand astfel relevanta sa continua in peisajul stiintific actual.

Bibliografie

1. Pancake Sorting in Python | GeeksforGeeks, accesată pe mai 25, 2025, https://www.geeksforgeeks.org/pancake-sorting-in-python/
2. Pancake sorting - Wikipedia, accesată pe mai 25, 2025, https://en.wikipedia.org/wiki/Pancake_sorting
3. Pancake Sort in Computer Algorithms | Siberoloji, accesată pe mai 25, 2025, https://www.siberoloji.com/pancake-sort-in-computer-algorithms-an-informative-guide/
4. www.comp.nus.edu.sg, accesată pe mai 25, 2025, https://www.comp.nus.edu.sg/~leonghw/uit2201/Fa2016/Lectures/L12/2016-07-Pancake-Flipping.pdf

5. helios2.mi.parisdescartes.fr, accesată pe mai 25, 2025, https://helios2.mi.parisdescartes.fr/~bouzy/publications/bouzy-pancake-cgw2015.pdf
6. "The Pancake Problems", accesată pe mai 25, 2025, http://dwest.web.illinois.edu/openp/pancake.html
7. Pancake Sorting Algorithm | Labuladong Algo Notes, accesată pe mai 25, 2025, https://labuladong.online/algo/en/frequency-interview/pancake-sorting/
8. Open Season: Pancake Flipping | The Aperiodical, accesată pe mai 25, 2025, https://aperiodical.com/2016/02/open-season-pancake-flipping/
9. Pancake Sorting - LeetCode, accesată pe mai 25, 2025, https://leetcode.com/problems/pancake-sorting/
10. www.cs.uni.edu, accesată pe mai 25, 2025, https://www.cs.uni.edu/~wallingf/teaching/cs3530/sessions/session20/bounds-for-sorting-by-prefix-reversal.pdf
11. TIL the only scientific paper ever published by Bill Gates was a possible solution to a mathematical problem about pancake flipping. : r/todayilearned - Reddit, accesată pe mai 25, 2025, https://www.reddit.com/r/todayilearned/comments/1eayjeq/til_the_only_scientific_paper_ever_published_by/
12. Engineering bacteria to solve the Burnt Pancake Problem - PMC, accesată pe mai 25, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC2427008/