

State of art pentru Echipa Sensor middleware

Bogdan Carina
Universitatea de Vest, Informatica aplicata

January 26, 2011

1 Senzor middleware

Middleware este software-ul care oferă o legătură între două aplicații soft diferite și pasează datele între ele. Middleware-ul oferă posibilitatea de a accesa datele aflate într-o bază de date să fie accesate printr-o alta. Se mai poate spune că middleware este software-ul de integrare de date. ObjectWeb definește middleware-ul ca fiind: "The software layer that lies between the operating system and applications on each side of a distributed computing system in a network."¹ În română : "Stratul software care stă între sistemul de operare și aplicații în fiecare parte a sistemului distribuit a rețelei. " Scopul proiectului Sensor Middleware este de a defini și implementa o arhitectură convenabilă care să răspundă cerințelor impuse de construirea unei rețele wireless de senzori care să fie capabili să detecteze, să înregistreze și să prelucreze date care vizează mărimi ale mediului cum ar fi: temperatura, curentul electric, consumul de gaz etc.

2 Arhitectura minimală

Abordarea proiectului nostru va avea în vedere trei probleme importante care constituie si nivelele arhitecturii modelului de senzor middleware:

1. Modelul hardware al senzorului

- construcția efectivă tehnică a senzorilor

¹Krakowiak, Sacha. "What's middleware?".

- topologia senzorilor adică modelele arhitecturare de amplasare a senzorilor în cadrul locuinței

2. Puntea între hardware și software

- dispozitivul de transformare a semnalelor analoge transmise de senzor în semnale binare adică date ce pot fi
- prelucrate ulterior de un soft specializat
- stocarea datelor într-o baza de date

3. Modelul software de prelucrare a datelor care presupune

- agregarea datelor
- serializarea datelor
- clusterizarea datelor
- generare de grafice
- compararea datelor
- detectarea punctelor de inflexiune
- detectarea erorilor cauzate de preluarea datelor

Abordarea middleware pentru rețele de senzori a fost propusă pentru clusterizarea datelor, consumul de energie și agregarea datelor și a mecanismului de colectare a acestora. Modelul middleware pentru Rețelele Wireless de Senzori ² poate fi abordat din două puncte de vedere: abordarea bazată pe bază de date sau abordarea bazată pe agent cea care satisface nevoile proiectului nostru. Rețeaua de senzori este un sistem reactiv care va răspunde schimbărilor de context bazate pe evenimente. Un agent este un cod special care se va plimba de-a lungul nodurilor ³ pentru a executa codul local pe baza evenimentului pe care îl va primi. Modelul Sensor Ware ⁴ folosește abordarea bazată pe agenți. În acest model interpretorul de date rulează pe fiecare nod care va detecta fiecare schimbare venită din exterior. O altă abordare bazată

²Wireless Sensor Network abreviat ca WSN

³Fiecare senzor care alcătuiește rețeaua de senzori va fi denumit generic nod

⁴Boulis, C.C. Han, and M. B. Srivastava. Design and Implementation of a Framework for Programmable and Efficient Sensor Networks. In MobiSys 2003, San Francisco, USA, May 2003.

pe modelul gent este Dfuse⁵ care este similară cu viziunile Sensor Ware. Aceste modele sunt implementate pe panouri embedded în special pentru probleme cum ar fi senzorii de lumină și temperatură.

3 Modele de construcție a senzorilor

Vom prezenta câteva modele de senzori de temperatură care sunt deja dezvoltate împreună cu avantajele și dezavantajele pe care le prezintă. În acest sens ne vom îndrepta atenția spre senzorii de temperatură din seria TE-6100. Porțiunea senzitivă a acestor senzori este reprezentată de elemente de nichel și silicon a căror rezistență variază odată cu schimbarea temperaturii. Modelele de senzori TE-6100 bazate pe fire de nichel sau tip de rezistență bazate pe silicon conferă o rezistență variată pentru un controller. Când temperatura senzorului se modifică, rezistența senzorului se modifică în conformitate. Senzorul va arăta o schimbare pozitivă de rezistență în concordanță cu temperatura. Senzorul de silicon se modifică 0.75% per grad Celsius cu o referință la rezistență de 1035 ohms la 256 grade Celsius. Senzorul de nichel se modifică 5.4 ohms per grad Celsius cu o referință la rezistență de 1000 ohms la 21 grade Celsius. Modelele asemănătoare din această categorie modelelor noastre de senzori sunt:

3.1 Modelul TE-6100-1

Acest model (figura de jos) este destinat pentru o paletă universală de aplicații, însă acuratețea și consumul resurselor nu sunt prea avantajoase.

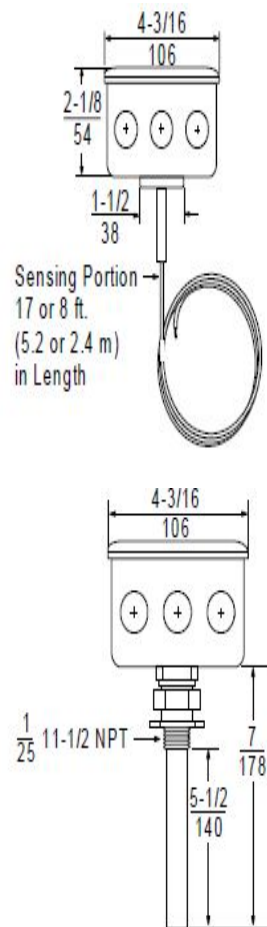
3.2 Modelul TE-6100-3

Acest model (figura de jos) este destinat pentru controlul temperaturii și memorarea acesteia. Modelul hardware include un element cu două cabluri de nichel, unul care se întinde de-a lungul a două cilindre concentrice și altul care este izolat termic și care este destinat producerii unui lag de timp la schimbări drastice de temperatură.

În modelul nostru

În modelul senzorului nostru schimbările drastice de temperatură nu vor

⁵R. Kumar, M. Wolenetz, B. Agarwalla, J. Shin, P. Hutto, A. Paul, and U. Ramachandran. DFuse: A Framework for Distributed Data Fusion. SenSys, 2003.



fi ignorate de hardware, ci înregistrate ca date, iar ulterior în procesul de prelucrare de date, pe baza unor marje de erori, vor fi excluse din calcul.

4 Cerințele arhitecturii pentru rețelele de senzori

Programabilitatea

în funcție de tipul de senzor -staționar sau mobil rețeaua care facilitează comunicarea între senzorul fizic efectiv și mașina care va prelua datele trebuie să îndeplinească anumite cerințe. Senzorii staționari vor comunica prin pro-

tocoale preexistente și caracteristice tipului de senzor ales, pe când senzorii mobili vor avea o rețea de comunicare ad-hoc. Trebuie luat în considerare și pericolul de deconectare a senzorului de la rețea, lungimea de bandă mică, limitarea aprovizionării cu energie a senzorilor, etc. în consecință rețeaua de senzori trebuie să suporte funcționalitate de programare, operații asincrone și o arhitectură bazată pe componente. Există asemenea limbaje orientate eveniment care ar putea fi folosite pentru situațiile în care apar operații de deconectare și anume: NesC ⁶ și galsC ⁷. Însă programarea fiecărui senzor este dificilă, mai ales dacă există rețele de senzori cu numeroase noduri. Noile paradigme de programare, asemenea celor abordate în proiectul nostru, își îndreaptă atenția spre funcționalități specifice la nivel de aplicație și nu la nivel de nod. Soluțiile middleware eficiente pot ascunde complexitatea în ceea ce privește configurarea individuală a nodurilor bazată pe capabilitățile arhitecturii software și hardware.

Adaptabilitatea

Multe aplicații de senzori au nevoie să își poată schimba comportamentul în funcție de starea mediului în care se află. De exemplu senzorii de lumină vor trebui să funcționeze în două moduri diferite și anume în regim de zi și de noapte. Acest lucru este natural deoarece senzorii nu vor trebui să detecteze lumina zilei ci doar acea lumină venită de la o sursă artificială. În cazul în care senzorul nu dispune de memorie suficientă pentru comutare de context, adică de regim de funcționare, atunci avem două posibilități de a rezolva această problemă. Prima ar fi că datele vor trebui trimise unui server pentru lăurarea acestei decizii și serverul va fi cel care va dicta senzorului regimul corespunzător de funcționare. A doua opțiune ar fi ca senzorul să aibă posibilitatea de a hotărî singur ce regim să urmeze urmînd să își downloadeze singur algoritmul necesar funcționării care este disponibil pe serverul de decizie. Deoarece am convenit să păstrăm luarea deciziilor și programarea la nivel de aplicație și nu la nivel de nod proiectul nostru va aborda prima variantă.

⁶D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D.Culler. The nesC Language: A Holistic Approach to Networked Embedded Systems. In Proceedings of Programming Language Design and Implementation (PLDI), June 2003.

⁷Elaine Cheong, Jie Liu galsC: A Language for Event-Driven Embedded Systems <http://ptolemy.eecs.berkeley.edu>

Scalabilitate

Majoritatea implementărilor de senzori profită de localizarea spațială. De exemplu, în cazul senzorilor de detectare a mișcării sau a temperaturii, informația provenită de la nodurile mai apropiate este evaluată și pe baza ei deciziile ulterioare vor fi făcute.

Dinamism moștenit

Noduri noi adăugate și eliminate din cuibul de senzori vor crea o schimbare în topologia senzorilor. De aceea trebuie să avem un mecanism pentru aplicație pentru a determina dinamic nodurile și servicii suport.

Priorități de timp real

Rețelele de senzor sunt utilizate pentru a monitoriza fenomene de timp real și nu alte informații stocate. Acest lucru pune problema semnării priorităților în timpul rulării și cum să menținem ordinea naturală de timp real. În rețelele de senzori prioritățile trebuie să fie explicit specificate. Prioritatea unui mesaj depinde de contextul sistemului. De aceea, prioritățile mesajelor trebuie să fie asignate la runtime de către middleware și ar trebui să fie bazate pe context. Managementul fuziunii senzorilor

Aplicațiile implicate în detectarea mișcării folosesc un număr mare de senzori pentru a identifica un obiect particular. Trebuie astfel întreținut un manager pentru fuziunea senzorilor care să suporte fuziunea evenimentelor de nivel jos (cum ar fi temperatura mică și nivele de lumină) pentru a prelucra evenimente de nivel înalt cum ar fi detectarea obiectelor folosind diverși algoritmi de fuziune. Acest manager trebuie să fie capabil să selecteze rolul cel mai bun pentru modelele disponibile și să aloce roluri nodurilor din rețea.

Manager de context

Aplicațiile cu senzori necesită adaptări în funcție de schimbările comportamentului și trebuie să se poată adapta la un context anume în care trebuie să lucreze. Contextul poate fi definit ca un set de stări de schimbare care vor conduce la o stare particulară a rețelei.

5 Achiziționarea datelor de la senzor

Achiziționarea datelor este procesul de acumularea semnalelor care măsoară condiții fizice din lumea reală și de conversie a rezultatelor preluate în valori digitale numerice care ulterior pot fi manipulate de calculator. Sistemele de achiziție a datelor de la senzori (abreviate DAS sau DAQ), în mod tipic, convertesc forme ondulatorii analogice în valori digitale menite prelucrării computerizate. Componentele unui astfel de sistem sunt:

- Senzorii care convertesc parametri fizici în semnale electrice
- Circuite de condiționări ale semnalelor pentru convertirea semnalelor de la senzori într-o formă care va putea fi convertită în valori digitale.
- Convertoare analog spre digital, care vor converti semnalele senzorilor în valori digitale

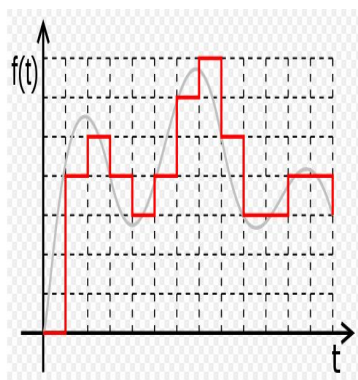
Aplicațiile de achiziție de date sunt controlate de programe soft dezvoltate folosind limbaje de programare diferite, în cazul nostru Java. Pentru a accesa și controla data preluată de hardware-ul de achiziționare se va putea folosi un API care va putea rula pe diverse sisteme de operare.

Achiziționarea datelor va începe cu fenomenul fizic sau proprietatea fizică care urmează a fi măsurată, cum ar fi temperatura, intensitatea luminii, consumul de energie electrică, consumul de gaz etc. Indiferent de mărimea fizică măsurată, starea fizică care urmează a fi măsurată trebuie să fie mai întâi transformată într-o formă unică care va putea fi preluată de sistemul de achiziție. Această transformare va fi realizată de senzorul efectiv.

Abilitatea de receptare de date a sistemului depinde de tipul de senzor care va detecta variația proprietății fizice respective. DAQ necesită numeroase tehnici de condiționare pentru semnale pentru a modifica numeroase semnale electrice în curent electric pentru a fi digitalizate utilizând Convertorul analog spre digital (Analog-to-digital converter (ADC)).

Semnalele pot fi digitale, numile semnale logice, sau analoage. Condiționarea semnalelor ar putea fi necesară dacă semnalele preluate nu sunt inteligibile pentru hardware-ul DAQ utilizat. Semnalul va trebui fie amplificat, filtrat, fie demodulat.

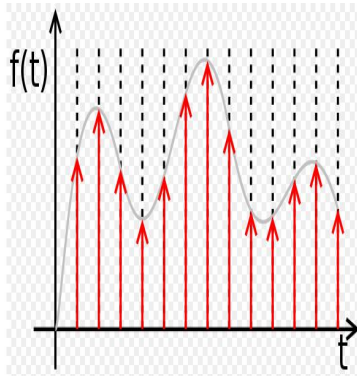
Conversia analog spre digital



Un convertor analog la digital (analog-to-digital converter ADC) este un dispozitiv care va converti o cantitate continuă la un număr digital discret. Tipic ADC-ul este un dispozitiv electronic care va converti un input analog de un anumit voltaj sau curent într-un număr digital proporțional cu mărimea care definește curentul sau voltajul. Output-ul digital folosește o numeroase scheme de codificare diferite. Un ADC poate fi folosit să măsoare valori izolate, însă în cazul nostru se vor folosi semnale dependente de timp și le va transforma în secvențe digitale. Rezultatul va fi o cantitate de date măsurată atât în timp, cât și în valoare. Semnalele pot fi încadrate în numeroase categorii, însă distincția cea mai comună se realizează între semnalele discrete și cele continue în funcție de funcțiile matematice peste care sunt definite. Semnale discrete de timp sunt deseori referite ca și serii de timp. Semnalele continue de timp sunt referite ca simplu semnale continue.

Un semnal discret sau un semnal discret de timp este o serie de timp care reprezintă o secvență de anumite cantități. Cu alte cuvinte, reprezintă o serie de timp care este o funcție definită peste un domeniu de numere discrete. Fiecare valoare a secvenței s e numește eșantion sau probă. Spre deosebire de semnalele continue, un semnal discret nu este o funcție de un argument continuu, chiar dacă se poate ca semnalul să fi fost obținut prin colectarea unui semnal continuu. Atunci când un semnal discret corespunde unor valori uniforme de timp, va exista o rată de colectare. Rata de colectare nu va fi prezentă în secvența de date, deci va trebui să s e atribuie o variabilă separată.

Semnalul digital utilizat va fi un semnal dintr-o serie discretă de timp pentru care amplitudinea și timpul sunt discrete și care va lua un singur set de valori discrete și de asemenea a input-ului și output-ului sunt discrete.



Seria de date utilizată va fi o secvență de puncte de date, măsurate la valori de timp succesive și uniforme. Intervalul nostru de timp între măsurători va fi constant de 6 secunde.

Serile de date au o ordine temporală naturală. Acest lucru le afce distincte de alte probleme de analiză a datelor în care nu există o ordine naturală a observărilor. Un model de serii de date va reflecta în general faptul că observările apropiate ca timp vor fi și mai apropiate decât observările realizate într-un timp viitor. și mai mult, modelele serilor de date vor face uz de ordonarea într-o singură direcție astfel încât valorile pentru o perioadă dată de timp va fi exprimată ca fiind derivată din valori din trecut și nu valori din viitor.

6 Prelucrări software ale serilor de date

6.1 Clusterizarea datelor provenite de la senzori

Analiza bazată pe clusterizare este procedeul de a găsi grupuri de similarități într-o serie de date. Deoarece evenimentele vizate pentru prelucrare sunt prea individuale, ele vor fi grupate în anumite grupuri bazate pe anumite trăsături similare. Un termen necesar aici este *clasificarea* procesul de asignare a unui nou item sau observație la locul său potrivit într-un set predefinit de clase de categorii.

Clusterizarea serilor de date cu variabile multiple are ca scop gruparea seturilor de date care au caracteristici similare. Aceste grupuri pot fi analizate ulterior în detaliu pentru a face descoperiri în miezul caracteristicilor comune a fiecărui grup de date.

Clusterizarea folosind factori de similaritate

Factorii de similaritate pot fi folosiți în locul distanței Euclidiene (abordare veche) pentru a măsura similaritatea între două seturi de date multivariate X_1 și X_2 . Krzanowski⁸ a dezvoltat o metodă pentru măsurarea similarității a două seturi de date folosind factorul de similaritate PCA care se va calcula folosind componenta k cea mai mare a fiecărui set de date. Componentele principale (PC) sunt vectorii matricei de covarianță a setului de date multivariabile. Factorul de similaritate S_{PCA} este definit ca :

$$S_{PCA} = \Delta \frac{1}{k} \sum_{i=1}^k \sum_{j=1}^k \cos^2 \theta_{ij}$$

unde k este numărul selectat din ambele seturi de date, θ_{ij} este unghiul dintre al i -lea PC a lui X_1 și al j -lea PC al lui X_2 . Numărul de PC-uri, k , poate fi ales astfel încât PC-ul k să descrie cel puțin 95% varianță în fiecare set de date.

Algoritmul propus este cel care folosește K mijloace de factori de similaritate

Fiind date Q seturi de date, $X_1, \dots, X_q, \dots, X_Q$, trebuie să fie clusterizate (încuibate) în K clustere.

1. Fie al j -lea set de date reprezentat în al i -lea set de date numit $X_j^{(i)}$. Se va computa setul de date agregat $X_i (i = 1, \dots, K)$, pentru fiecare din cele K clustere ca fiind:

$$X_i = [(X_1^{(i)})^T \dots (X_j^{(i)})^T \dots (X_{Q_i}^{(i)})^T]^T$$

unde Q_i este numărul de seturi de date în X_i . Să se consemneze că $\sum_{i=1}^K Q_i = Q$.

2. Se va calcula disimilaritatea între setul de date X_q și fiecare din cele K clustere agregate $X_i, i = 1, \dots, K$ ca fiind:

$$d_{i,q} = 1 - SF_{i,q}$$

⁸Oxford Statistical Science Books, W. Krzanowski, Principle of Multivariate Analysis

unde $SF_{i,q}$ este factorul de similaritate între al q-lea set de date și al i-lea cluster. Apoi se va lăsa setul de date agregat X_i să fie setul de date de referință. Setul de date X_q va fi asignat cluster-ului căruia îi este cel mai puțin nesimilar și anume clusterul cu valoarea cea mai mică a lui $d_{i,q}$. Se va repeta pasul pentru toate Q cluster.

3. Se va calcula media nesimilarităților a fiecărui set de date după formula:

$$J(K) = \frac{1}{Q} \sum_{i=1}^K \sum_{X_q \in X_i} d_{i,q}$$

4. Dacă valoarea lui $J(K)$ s-a schimbat dinaintea primei iterații atunci ne vom întoarce la pasul 2, altfel stop.

6.2 Compararea seriilor de date pentru sisteme de senzori

În statistici, corelația și dependența sunt elemente dintr-o clasă largă de relații statistice între două sau mai multe variabile aleatoare sau de date, de valori observate. Există două metode de comparare a serilor de date după cu urmează: Măsura cea mai cunoscută de dependență între două cantități este coeficientul Pearson sau "de corelație Pearson". Acesta se obține prin împărțirea covarianței dintre cele două variabile la produsul abaterilor standard. Coeficientul de corelație a populației $\rho_{X,Y}$ dintre două variabile aleatoare X și Y cu valorile așteptate μ_X și μ_Y și abaterile standard σ_X și σ_Y , este definit astfel:

$$\rho_{X,Y} = \text{corr}(X, Y) = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}$$

unde E este valoarea așteptată, cov înseamnă covarianță și corr fiind o notație alternativă utilizată pe scară largă pentru corelație Pearson. Corelația Pearson este definită numai dacă ambele deviații standard sunt finite și ambele dintre ele sunt nenule. Este un corolar al inegalității Cauchy-Schwarz deoarece această corelație nu poate depăși 1 în valoare absolută. Coeficientul de corelație este simetric: $\text{Corr}(X, Y) = \text{Corr}(Y, X)$.

Corelația Pearson este 1 în cazul unei relații liniare perfecte pozitive (în creștere), -1 în cazul unei relații liniare perfecte negative (de scădere) și are

valori cuprinse între -1 și 1 în toate celelalte cazuri care indică gradul de dependență liniară între variabile. Cu cât coeficientul este mai aproape de a fi -1 sau 1, cu atât mai puternică este corelația dintre variabile.

Coeficientul Pearson utilizat în compararea seriilor de date pentru sisteme de senzori

Fiind date serii de timp care reprezintă consumul de curent electric se dorește o comparare low-end a utilizatorilor sau respectiv a claselor de utilizatori. Pentru aceasta ne putem folosi de coeficientul Pearson de dependență liniară (deoarece graficele de consum sunt compuneri de funcții liniare). Pentru a aplica coeficientul Pearson asupra unor seturi de date, acestea trebuie întâi normalizate.

Prin normalizarea seturilor de date se înțelege:

1. Ambele seturi să aibă același număr de elemente
2. Ambele seturi să fie reprezentate pe aceeași scară
3. Eliminarea Zgomotului

Algoritmul pentru generarea coeficientului Pearson

1. Vom avea în vedere două seturi de date S_1 și S_2 , abaterea standard a celor două seturi d_x și d_y și medile aritmetice m_1 și m_2 ale valorilor fiecărui set de date se va calcula abaterea standard a setului 1 de date apoi a setului 2 de date conform formulei:

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2}$$

unde σ este abaterea standard, N este mărimea setului de valori, x_i este valoarea considerată, μ este media valorilor setului.

2. Apoi se va calcula media aritmetică a tuturor valorilor setului 1 de date, respectiv setul 2 de date vom lua ca iterator o variabilă i care va începe la 0 și se va incrementa cu o unitate cât timp variabila i va fi mai mică decât numărul de valori din setul de date la fiecare pas se va calcula raportul dintre diferența dintre valoarea extrasă din set și

media setului respectiv și deviația standard a setului. Acest procedeu se va repeta pentru fiecare set. La fiecare pas vom avea o variabilă rezultat care va păstra radicalul produsului operației calculate pentru setul 1 respectiv setul 2.

3. La sfârșit se va face împărțirea unității la produsul sumei rezultatelor de la fiecare pas cu mărimea setului în cauză.

A doua metodă se numește **Circle Matching**

Circle matching se bazează pe algoritmi primitivi de comparare de imagini.

Algoritmul utilizat primește ca date de intrare 2 grafice de consum ce reprezintă seturile de date de comparat și returnează o valoare între 0 și 100 ce reprezintă procentajul de similaritate. Algoritmul funcționează pe următoarele principii:

Fie graficul de consum A roșu și graficul de consum B albastru Circle matching va detecta pentru fiecare punct al graficului A- puncte albastre în proximitatea sa (proximitate determinată de o constantă alfa, aleasă în funcție de mărimea setului de date)

Fiecare punct ce îndeplinește condiția de proximitate incrementează un contor; Algoritmul returnează valoarea: $Contor/NumărTotalPuncte(A+B)$

6.3 Detectarea anomaliilor din fluxul de date a senzorilor de mediu

Unii senzori operează în condiții grele, iar datele trimise de aceștia către rețelele de comunicare, pot deveni eronate. Senzorii insitu sunt acei senzori care sunt localizați în mediul în care monitorizează. Tipurile de erori pot fi cauzate chiar de aceștia, de transmiterea unor date eronate, sau de comportamentul ocazional al sistemului, lucruri care sunt de interes pentru comunitățile de știință. Așadar avem nevoie de o asigurare și un control în mod automat a calității datelor(QA/QC). Acest lucru este foarte util atunci când dorim să facem o analiză mai detaliată a datelor, care la prima vedere pot părea anormale sau pentru care trebuie să executăm cu totul alte acțiuni(de exemplu cazul unui dezastru natural). Aplicația cere ca datele anormale să fie verificate în timp aproape real, deci trebuie să fie rapid și efectuat incremental pentru a putea ține pasul cu rata colectării de date.

Pentru această metodă, se propune un senzor de simulare, ale cărui măsurători vor fi comparate cu măsurătorile senzorului actual. Astfel, vom clasifica o dată ca fiind normală sau anormală pe baza diferenței dintre anticipările modelului și măsurătoarea senzorului. Acest studiu dezvoltă o metodă de detecție a anomaliilor în timp real, care angajează un model de data-driven unidimensional a fluxului de date și un interval de predicție (PI), calculat dintr-un istoric recent.

Datele vor fi clasificate ca fiind normale în funcție de aserțiunea că rezultatul va fi PI sau nu. Această metodă nu cere niciun exemplu preclasificat de date, ci se potrivește unui volum larg de date și permite o evaluare de date incrementală rapidă, din momentul în care devine disponibilă.

Algoritmul pentru detectarea anomaliilor din serile de date

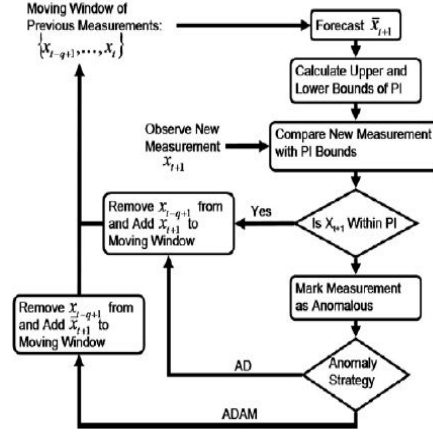
Acest studiu propune o metodă de redundanță analitică pentru detectarea anomaliilor, care folosește o fereastră mobilă de q măsurători ale senzorilor (sau valorile pe care le așteptăm de la aceștia):

$$D^t = x_{t-q+1}, \dots, x_t$$

pentru a clasifica următoarele secvențe de măsurători. Măsurătoarea va fi clasificată ca fiind anormală dacă deviază semnificativ de la valoarea anticipată făcută cu un pas înainte. Apoi metoda umple geamul cu q -ul cel mai recent și se continuă clasificarea cu următoarea măsurătoare primită de la senzor. Avem următorii pași începând de la timpul t :

Înainte valorilor calculate folosind D^t ca dată de intrare. Apoi metoda umple geamul cu q -ul cel mai recent și se continuă clasificarea cu următoarea măsurătoare luată de la senzor. Avem următorii pași (începând de la timpul t):

1. Metoda de anticipare cu un pas înainte ia ca input D^t pentru a anticipa x_{t+1} valoare așteptată de la senzorul măsurat la timpul $t + 1$;
2. Calculează legăturile din intervalul de anticipare cu probabilitatea p , realizând un rang
3. Comparăm măsurarea de la timpul $t + 1$ cu (2) . Dacă datele nu aparțin intervalului rangului de la (2) atunci sunt considerate ca fiind anormale.



4. (a) Sub strategia de detecție a anomaliilor si a atenuării (ADAM), dacă măsurătoarea este clasificată ca fiind anormală modifică D^t ștergând x_{t-q+1} din spatele ferestrei și adăugând x_{t+1} în fața ferestrei pentru a crea D^{t+1}
5. (b) Dacă măsurătoarea este clasificată ca fiind normală, modifică D^t ștergând x_{t-q+1} din spatele ferestrei și adăugând x_{t+1} în fața ferestrei pentru a crea D^{t+1} .
6. repeta pașii 1-4.

6.4 Numărarea elementelor distincte dintr-un flux de date

În bazele de date, o problemă fundamentală o reprezintă numărarea elementelor diferite dintr-un tabel. Avem nevoie de un algoritm de optimizare a datelor primite într-un anumit timp și spațiu. Acest algoritm este considerat eficient dacă folosește foarte puțin spațiu, dacă are unul sau cât mai puține cicluri și dacă procesează fiecare element introdus într-un timp cât mai scurt. Presupunem că avem un șir de elemente a_1, \dots, a_n , n aparține domeniului $|m| = 1, \dots, m$. $F_0 = F_0(a)$ (cu alte cuvinte momentul frecvenței 0) reprezintă numărul elementelor distincte care apar în secvență. Algoritmul care aproximează F_0 este considerat eficient dacă folosește mai mulți $(1/\epsilon, \log n, \log m)$ biți de memorie, unde $1 \pm \epsilon$ este un factor în cadrul căruia F_0 trebuie să fie aproximat.

Algorithm	Spațiu	Timp/element
1, Thm. 1	$O(1/\epsilon^2 \cdot \log m)$	$\sim O(\log m)$
2, Thm. 2	$\sim O(1/\epsilon^2 + \log m)$	$\sim O(1/\epsilon^2 \cdot \log m)$
3, Thm. 3	$\sim O(1/\epsilon^2 + \log m)$	$\sim O(\log m)$ [amortizat]

Vom prezenta trei algoritmi cu diferite echilibrări timp-spațiu pentru aproximarea lui F_0 . Algoritmii vor fi condiționați de ϵ și $\log m$ (surprimând dependența de n). de exemplu dacă $m < n$ este mult mai avantajos să avem algoritmi ai căror legături depind de $\log m$ și nu de $\log n$, iar dacă $m > n$ putem realiza un mic truc utilizând o dispersie a datelor, care reduce descrierea fiecărui element în curs ($O(\log n)$). De asemenea presupunem că există $\epsilon_0 < 1$ în așa fel încât acuratețea parametrului ϵ dat algoritmului este cel mult ϵ_0 .

Algoritmul 1

Acest algoritm alege la întâmpalre o funcție $h : |m| \rightarrow [0, 1]$. Apoi aplică funcția la fiecare element din a și menține valoarea $v = \min_j h(a_j)$. În final estimarea va fi $F_0 = 1/v$. Apoi, mai luăm o funcție $h : |m| \rightarrow [0, 1]$, dar menținem elementele a_i la $t = O(1/\epsilon^2)$ pe care funcția h le va evalua la cea mai mică valoare t . Estimarea $F_0 = 1/v$, v fiind valoarea t cea mai mică.

Teorema 1 Avem un algoritm care pentru oricare $\epsilon, \delta > 0, (\epsilon, \delta)$ aproximează F_0 folosind $O(1/\epsilon^2 * \log m * \log(1/\delta))$ biți de memorie și $O(1/\epsilon^2 * \log m * \log(1/\delta))$ timp de procesare per element.

Demonstrație: Luăm o pereche independentă de funcție $h : |m| \rightarrow [0, 1]$, unde $M = m^3$ este injectivă pe elementele lui a . Cu probabilitatea cel puțin $1 - 1/m$.

Algoritmul 2 Scopul acestui algoritm este să definească o cantitate care poate fi aproximată într-un model de flux de date și care poate fi folosit pentru aproximarea lui F_0 .

Teorema 2 Avem un algoritm care pentru oricare $\epsilon, \delta > 0, (\epsilon, \delta)$ aproximează F_0 folosind $O(1/\epsilon^2 * \log m * \log(1/\delta))$ biți de memorie și $O(1/\epsilon^2 * \log m * \log(1/\delta))$ timp de procesare per element.

Demonstrație: Avem b_1, \dots, b_{F_0} care denotă F_0 elemente distincte din

fluxul a , iar B denotă setul b_1, \dots, b_{F_0} . Considerăm o funcție complet întâmplătoare $h > |m| - > |R|$ și definim $r = Pr_h[h^{-1}(0) \cap B \neq null]$. Mai întâi arătăm că dacă R și $!!$ sunt constante multiple una pentru cealaltă, atunci aproximând r -ul este un bun mod de a aproxima F_0 .

6.5 Aproximare de grafic detectarea punctelor critice

Interpolarea lineară este o metodă de a potrivi graficele generate ca și curbe în mod normal folosind polinoame lineare. Este o metodă des utilizată în matematică, mai ales analiza numerică și numeroase aplicații mai ales se bazează pe grafice o folosesc. Interpolarea lineară este o formă simplă de interpolare.

Interpolarea lineară între două puncte date Fie punctele date prin coordonatele (x_0, y_0) și (x_1, y_1) , interpolarea lineară este linia dreaptă trasată între aceste puncte. Pentru o valoare x din intervalul (x_0, x_1) , valoarea lui y de-a lungul liniei drepte este dată de ecuația:

$$\frac{y - y_0}{x - x_0} = \frac{y_1 - y_0}{x_1 - x_0}$$

care poate fi derivată geometric ca și în imaginea de mai jos. Rezolvarea acestei ecuații în y , care este valoare necunoscută în x , ne dă:

$$y = y_0 + (x - x_0) \frac{y_1 - y_0}{x_1 - x_0} = \frac{(x - x_0)y_1 + (x_1 - x)y_0}{x_1 - x_0}$$

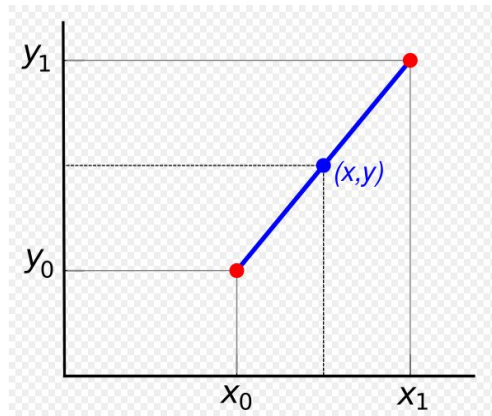
care este formula interpolării lineare din intervalul (x_0, x_1) . În afara acestui interval formula este identică cu extrapolarea lineară.

Interpolarea unui set de date

Interpolarea unui set de puncte de date $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ este definită ca fiind concatenarea interpolării lineare între fiecare pereche de puncte a setului de date. Rezultatul va fi o curbă continuă, cu o derivată discontinuă, totuși de clasă diferită C^0 .

Deseori interpolarea lineară este folosită pentru aproximarea unei valori a unei funcții f folosind două valori cunoscute a acelei funcții la fiecare din cele două puncte. Eroarea acestei aproximări este definită ca:

$$R_T = f(x) - p(x)$$



unde p denotă interpolarea lineară polinomială definită mai jos:

$$p(x) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0)$$

Se poate defini folosind teorema lui Rolle ⁹ că dacă f are o a doua derivată continuă, eroarea este limitată în:

$$|R_T| \leq \frac{(x_1 - x_0)^2}{8} \max_{x_0 \leq x \leq x_1} |f''(x)|$$

După cum s e vede aproximarea dintre cele două puncte pe o funcție definită devine din ce în ce mai rea cu cea de-a doua derivată a funcției care este aproximată.

⁹Studiu aprofundat la: <http://en.wikipedia.org/wiki/Rolletheorem>

