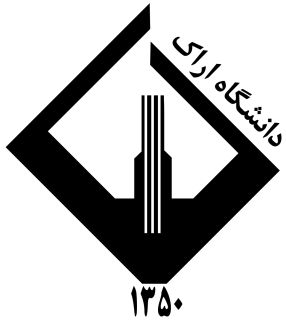


عنوان ارائه:

یک چارچوب تشخیص و ردیابی داده‌های پرت در داده‌های جریانی

PROUD: PaRallel OUtlier Detection for streams

توسط: علیرضا صادقی نسب
استاد: دکتر حسین غفاریان

تاریخ ارائه: ۱۳۹۹/۹/۲۶

مقدمه

- اطلاعات مقاله

  - عنوان: *PROUD : PaRallel OUtlier Detection for streams*
  - ماه و سال چاپ: *June* 2020
  - تعداد ارجاع: 1
  - عنوان کنفرانس:

Proceedings of the 2020 *ACM SIGMOD International Conference on Management of Data*

مقدمه

- اطلاعات نویسندگان

Theodoros Toliopoulos

PhD candidate, Aristotle University of Thessaloniki
Verified email at csd.auth.gr
Big data stream outlier detection

Cited by	All	Since 2015
Citations	20	20
h-index	2	2
i10-index	1	1

Anastasios Gounaris

Associate Professor of Aristotle University of Thessaloniki
Verified email at csd.auth.gr - Homepage
Informatics Computer Science

Cited by	All	Since 2015	VIEW ALL
Citations	1886	773	
h-index	25	13	
i10-index	44	19	

Apostolos N. Papadopoulos

Associate Professor, Department of Informatics, Aristotle University of Thessaloniki, Greece
Verified email at csd.auth.gr - Homepage
Databases Data Mining Big Data Analytics

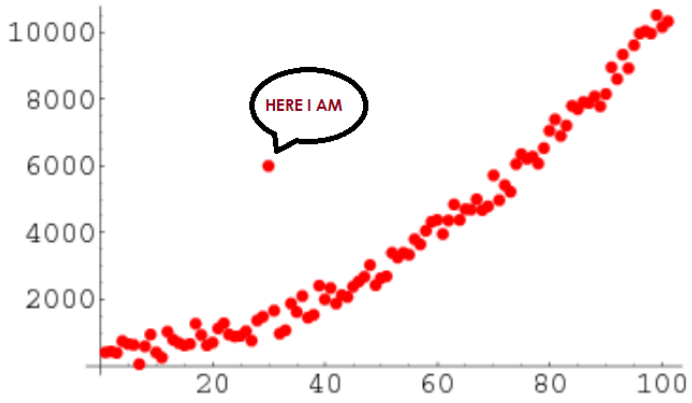
Cited by	All	Since 2015	VIEW ALL
Citations	3332	1339	
h-index	30	20	
i10-index	68	41	

فهرست مطالب

- مقدمه
- معرفی چارچوب
- پیکربندی و اجرای چارچوب
- بررسی نقاط قوت و ضعف

مقدمه

- داده‌های پرت
  - مجموعه نقاطی هستند فاصله معنادار و غیرعادی از سایر نقاط جمعیت دارند
  - البته باید توجه داشت که تعریف بالا، لزوماً دال بر پرت بودن داده نیست، تصمیم‌گیری این موضوع در نهایت بر عهده تحلیلگر داده یا یک پردازش مرتبط است.
  - روش‌های معمول تشخیص داده‌های پرت:
    - مبتنی بر فاصله
    - مبتنی بر چگالی مانند الگوریتم LOF
  - حوزه‌های کاربرد:
    - آمار
    - پردازش سیگنال
    - اقتصاد و امور مالی
    - داده‌کاوی و شبکه



مقدمه

- روش‌های تشخیص داده‌های پرت
  - ☆ روش‌های باناظر: به عنوان یک مسئله طبقه‌بندی، مدل می‌شود
    - ☆ چالش۱: رده‌ها نامتعادل هستند، جمعیت داده‌های پرت بسیار کوچکتر است
    - ☆ چالش۲: معیار *recall* مهمتر از معیار *precision* است
  - ☆ روش‌های بدون ناظر: به عنوان یک مسئله خوشه‌بندی، مدل می‌شود
    - ☆ چالش۱: داده‌های معمولی ممکن است یک الگو قوی نداشته باشند اما داده‌های پرت در یک منطقه کوچک، یک مشابهت زیادی را ایجاد کنند
    - ☆ چالش۲: در این گونه روش‌ها، اغلب، میزان *fp* ها زیاد است.
  - ☆ روش‌های نیمه ناظر: در این گونه روش‌ها، با داده‌های معمولی برچسب‌دار، داده‌های غیربرچسب‌دار تخمین زده می‌شوند تا یک مدل یادگیری شود. داده‌هایی که مناسب مدل داده‌های معمولی نباشند، در دسته داده‌های پرت قرار می‌گیرند

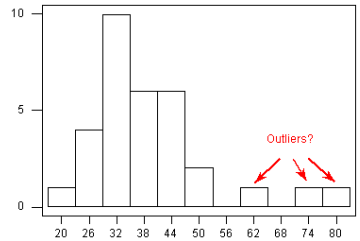
مقدمه

- روش‌های تشخیص داده‌های پرت (ادامه)

☆ روش‌های آماری

☆ روش‌های پارامتری: در این روش‌ها، فرض شده است که داده‌های معمولی به وسیله یک توزیع پارامتری θ تولید می‌شوند. عمده عیب این روش‌ها این است که به شدت به توزیع داده‌ها حساس هستند. برای مثال: $x^2 = \sum_{i=1}^{n} \frac{(o_i - E_i)^2}{E_i}$

☆ روش‌های غیرپارامتری: در این روش‌ها، فرض‌های کمتری در مورد داده ورودی انجام می‌شود. یک روش پرطرفدار این دسته، هیستوگرام است. اگر یک داده در یکی از بلوک‌های هیستوگرام قرار بگیرد، داده معمولی است در غیر این صورت، داده پرت تلقی خواهد شد. عمده عیب این روش، سختی انتخاب اندازه بلوک‌ها می‌باشد



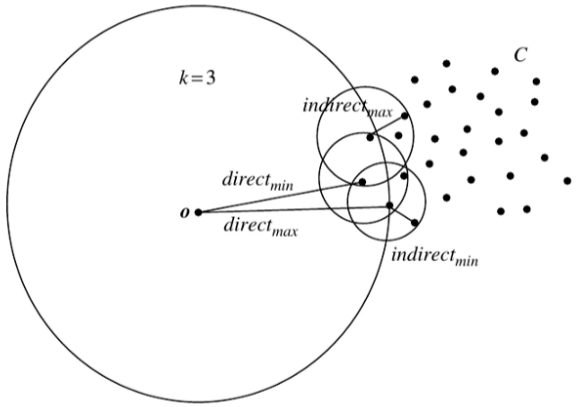
مقدمه

- روش‌های تشخیص داده‌های پرت (ادامه)

☆ روش‌های مبتنی بر مجاورت

☆ روش‌های مبتنی بر فاصله: یک داده، داده پرت است اگر در همسایگی او، به اندازه کافی نقاط دیگر در دسترس نباشند

☆ روش‌های مبتنی بر چگالی: یک داده، داده پرت است اگر تراکم آن، به نسبت کمتر از همسایگان آن باشد



مقدمه

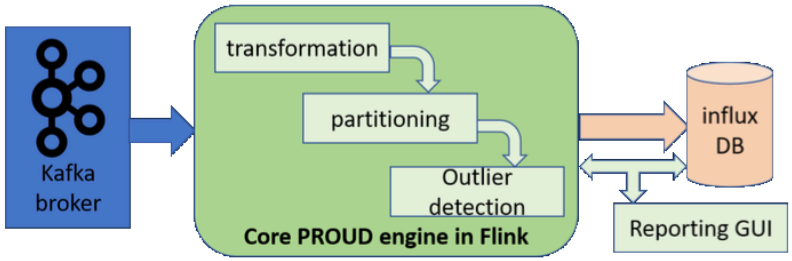
- داده‌های جریانی
  - داده‌هایی هستند که به طور مداوم توسط منابع مختلف تولید می‌شوند
  - در مقابل داده‌های ایستا، این نوع داده‌ها، پویا و نامحدود تلقی می‌شوند
  - پردازش این داده‌ها بدون دسترسی به همه داده‌ها صورت می‌گیرد
  - نمونه‌ای از کاربردها:
    - سفارش‌گیری در بازار سهام، بروزرسانی بلادرنگ دارائی بعد از تحرکات بازار
    - جمع‌آوری اکشن‌های کاربران بازی‌های ویدئویی آنلاین و نمایش فعالیت آن‌ها

مقدمه

- آپاچی کافکا
  - ☆ پلتفرم متن‌باز توزیع شده پردازش جریان‌های رخدادی
  - ☆ مقیاس‌پذیری، گذردهی بالا، مخزن دائمی ذخیره‌سازی رخدادها و دسترس‌پذیری بالا از مهمترین مزیت‌های آپاچی کافکا است
- آپاچی فلینک
  - ☆ چارچوب متن‌باز و موتور پردازشی توزیع شده برای محاسبات داده‌های متناهی و نامتناهی
  - ☆ مقیاس‌پذیری، تحمل خطای بالا، تاخیر پردازشی کوتاه و دسترس‌پذیری بالا از مهمترین مزیت‌های آپاچی فلینک است
- اینفلوکس‌دی‌بی
  - ☆ پایگاه‌داده سری زمانی متن‌باز
  - ☆ برای مصارف در لحظه بهینه‌سازی شده است؛ برای مثال ابزارهای تحلیل بلادرنگ و ذخیره‌سازی اطلاعات سنسورهای اینترنت اشیاء

معرفی چارچوب

- توصیف سیستم
  - ☑ آپاچی کافکا: برای مدیریت داده‌های جریانی ورودی به کار برده می‌شود
  - ☑ آپاچی فلینک: برای پردازش داده‌های جریانی ورودی به کار برده می‌شود
  - ☑ اینفلوکس‌دی‌بی: نتایج الگوریتم تشخیص داده پرت و فراداده‌های مربوط به عملکرد را ذخیره می‌کند
  - ★ معماری چارچوب پیشنهادی:



معرفی چارچوب

- بخش‌های اصلی سیستم

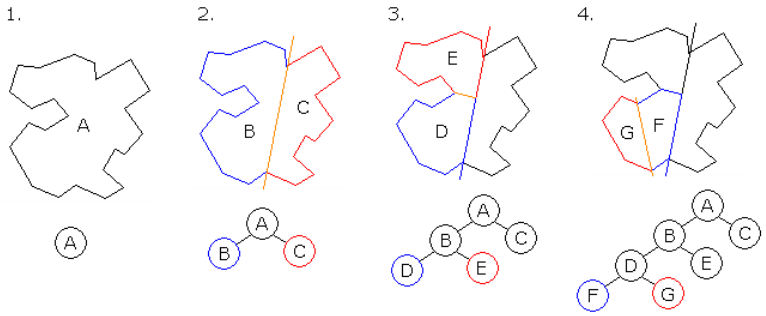
- ☑ انتقال داده‌ها: داده‌های ورودی به صورت پیوسته از مجاری ورودی و تبدیل آن به مدل مورد نظر
- ☑ افراز داده‌ها: داده‌ها به وسیله یک فناوی تعریف شده کاربر، از هم جدا شده و در قطعات جداگانه‌ای قرار می‌گیرند. لازم به ذکر است افراز بر پایه مقدار داده‌ها و پنجره‌بندی داده‌ها بر پایه برچسب زمانی آن‌ها صورت می‌گیرد
- ☑ تشخیص داده‌های پرت: الگوریتم انتخاب شده، به ازای هر قطعه اجرا می‌شود. از آنجایی که هر قطعه به صورت کاملا مستقل و موازی الگوریتم را اجرا می‌کند، امکان تولید مقادیر مثبت کاذب و منفی کاذب وجود نخواهد داشت
- ☑ ذخیره‌سازی و گزارش‌دهی: نتایجی که از هر قطعه به دست می‌آید، جهت ذخیره‌سازی، گزارش و پردازش‌های بعدی به سمت انباره هدایت می‌شوند

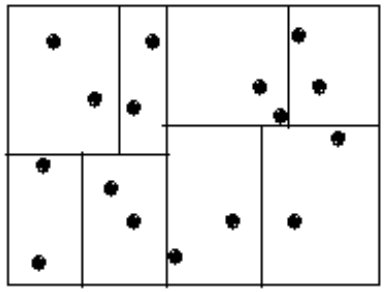
معرفی چارچوب

- انتقال داده:
  - هر داده‌ای که وارد می‌شود، سه مولفه اصلی دارد:
    - مقدار (عدد حقیقی)
    - برچسب زمانی ورود
    - شناسه یکتا
  - مدل داده‌ای پیاده شده کاملاً مستقل از الگوریتم تشخیص است و اگر الگوریتم مورد نظر نیاز به مشخصه‌های دلخواه دیگر باشد، به آسانی قابل گسترش و وفق‌پذیری است

معرفی چارچوب

- افراز داده:
  - وظیفه این بخش، تخصیص هر داده به یک قطعه منطقی است
  - افرازبندی در چارچوب پیشنهادی از ۲ طریق امکانپذیر است:
    - بر پایه شبکه مشبک: از فاصله اقلیدسی برای جداسازی نقاط از یکدیگر و انتقال آن‌ها به سلول‌های مشبک بهره می‌برد
    - بر پایه درخت: متکی بر ساخت درخت تقسیم‌کننده از داده‌های نمونه می‌باشد





معرفی چارچوب

- تشخیص داده‌های پرت:
  - بعد از اینکه داده‌ها، براساس مقدار افراز و براساس برچسب زمانی، در پنجره قرار گرفتند، الگوریتم تشخیص به ازای هر پنجره اجرا می‌شود.
  - هیچگونه ارتباطی بین الگوریتم تشخیص و افراز وجود ندارد و کاملا مستقل از هم هستند
- انتقال داده‌های پرت به خروجی:
  - انتقال داده‌های پرت هر قطعه به صورت مجزا انجام می‌شود
  - در انتها، معیارهای تعداد پنجره‌ها، زمان کل پردازش و میانگین زمان پردازشی هر قطعه محاسبه شده و در یک رابط کاربری به نمایش گذاشته می‌شوند

معرفی چارچوب

★ الگوریتم‌های پشتیبانی شده:

☆ روش‌های تک پارامتره:

- ✤ *naive*
- ✤ *advanced*
- ✤ *slicing*
- ✤ *advanced_extended*
- ✤ *mcod*
- ✤ *omcod*

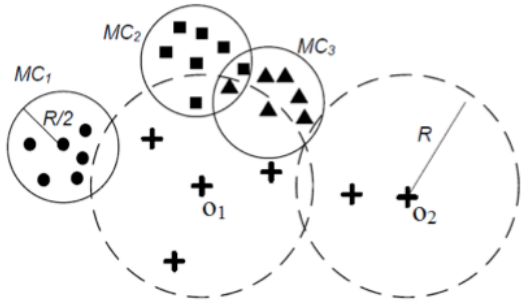
☆ روش‌های چند پارامتره:

- ✤ *amcod*
- ✤ *sop*
- ✤ *psod*
- ✤ *pmcsky*

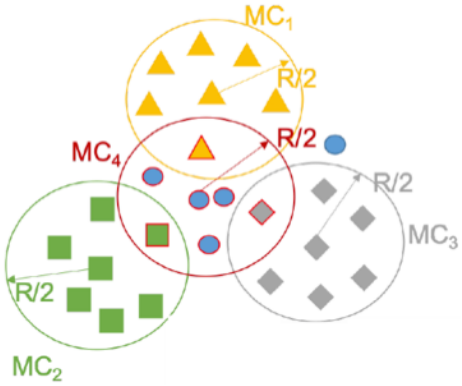
معرفی چارچوب

★ روش‌های تک پارامتره:

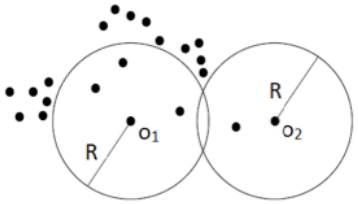
✤ *mcod*



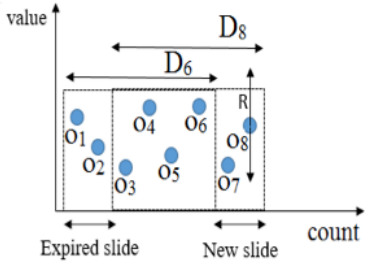
✤ *omcod*



✤ *umcod*



(a) Static Dataset

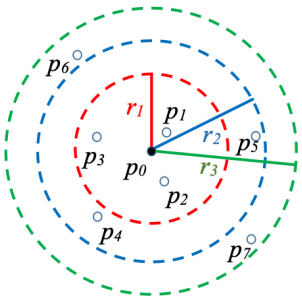


(b) Data Stream

معرفی چارچوب

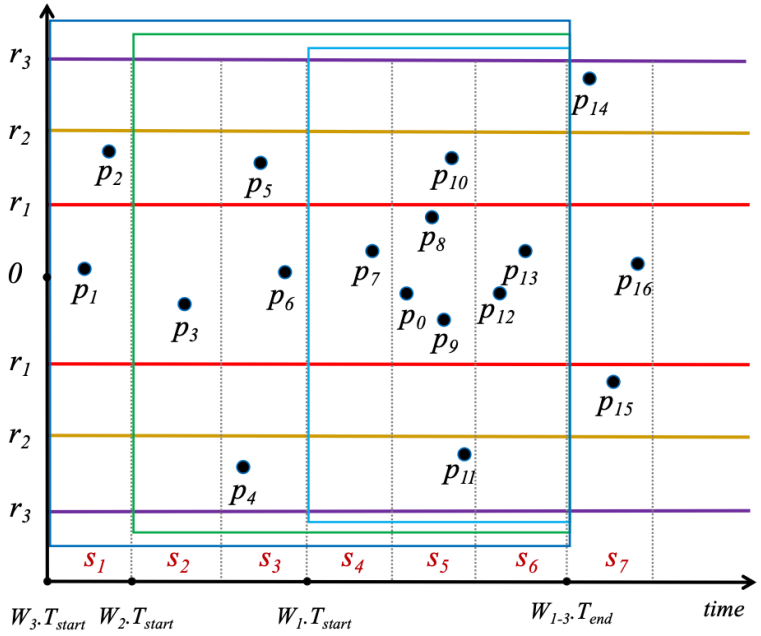
★ روش‌های چند پارامتره:

- ✤ *amcod*
- ✤ *psod*



Distance Interval	Neighbor Set
$[0,r_1]$	p_1,p_2,p_3
$(r_1,r_2]$	p_4,p_5
$(r_2,r_3]$	p_6,p_7

Dist. Int.	s_1	s_2	s_3	s_4	s_5	s_6	s_7
$[0, r_1]$	p_1	p_3	p_6	p_7	p_8, p_9	p_{12}, p_{13}	p_{16}
$(r_1, r_2]$	p_2		p_5		p_{10}		p_{15}
$(r_2, r_3]$			p_4		p_{11}		p_{14}



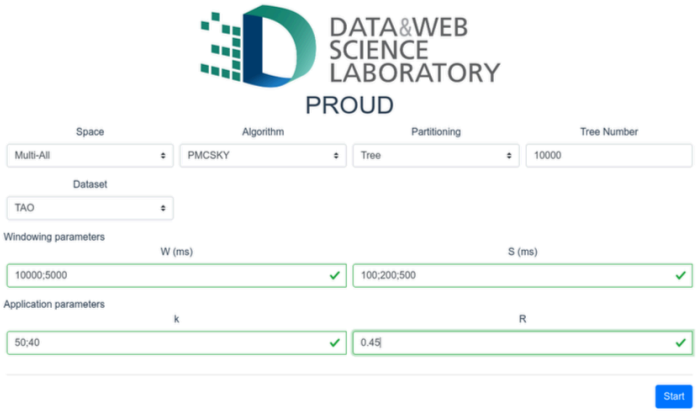
معرفی چارچوب

مزیت‌های توسعه‌پذیری:

- الگوریتم تشخیص: هسته اصلی چارچوب براساس الگوریتم‌های مبتنی بر فاصله پیاده شده است اما امکان افزودن الگوریتم‌های جدید به آسانی میسر است. هر الگوریتم، حالت، مدل و متغیرهای داخلی خود را دارد
- روش افراز: روش‌های افراز مبتنی بر درخت و شبکه مشبک به صورت پیشفرض در چارچوب وجود دارند اما امکان افزودن روش‌های دلخواه نیز وجود دارد
- امکان تعریف منبع داده و پایگاه داده متفاوت: به صورت پیشفرض، آپاچی کافکا به عنوان منبع داده و اینفلوکس‌دی‌بی به عنوان پایگاه داده، تعریف و پیکربندی شده‌اند. اما صرف نظر از سایر قسمت‌های چارچوب، می‌توان منابع دیگری را تعریف نمود

پیکربندی و اجرای چارچوب

- مراحل پیکربندی:
  - تولید مجموعه‌داده: برپایه دو مجموعه‌داده *Stock* و *Tao* و براساس توزیع زنگوله، مجموعه‌داده‌ها ساخته می‌شوند. بازه مقادیر داده‌ها با مجموعه‌داده‌های اصلی یکسان است
  - انتخاب نوع الگوریتم و الگوریتم: یک پارامتره باشد یا چند پارامتره
  - انتخاب روش افراز: برپایه درخت باشد یا شبکه مشبک
  - انتخاب پارامترهای دیگر ورودی چارچوب مانند: اندازه پنجره‌ها، میزان شعاع همسایگی و غیره



بررسی نقاط قوت و ضعف

- نقاط قوت
  - مقیاس‌پذیر بودن و توانا در مدیریت جریان حجیم داده‌های ورودی
  - کاملاً ماژولار بودن؛ جدا و مستقل بودن کامپوننت‌های چارچوب
  - امکان تعریف الگوریتم جدید تشخیص داده پرت، روش افراز، منابع داده و پایگاه‌داده‌های گوناگون به صورت مستقل
  - در دسترس بودن کد چارچوب به صورت عمومی
- نقاط ضعف
  - عدم پشتیبانی از داده‌های غیرعددی مانند رشته‌ها
  - عدم پشتیبانی از روش‌های تشخیص داده‌های پرت مبتنی بر چگالی

با تشکر از توجه شما