ارزیابی الگوریتم‌های کنترل هم‌روندی WW و WD برای مدیریت پایگاه داده‌ها، از طریق مدل‌سازی با پتری رنگی

فاطمه سعادت‌جو [۱،*]، میدیا بهزادیان [۲]، محمدعلی سعادت‌جو [۳]

اطلاعات مقاله

دریافت مقاله: ۱۳۹۴/۰۵/۰۸
پذیرش مقاله: ۱۳۹۵/۰۱/۲۹

واژگان کلیدی:
کنترل هم‌روندی،
شبکه پتری رنگی،
منتظرگذاشتن-میراندن،
زخمی‌کردن-منتظرگذاشتن،
ارزیابی،
پیشگیری از بن‌بست.

چکیده

اجرای هم‌روند تراکنش‌ها در پایگاه داده، ممکن است منجر به ناسازگاری شود. ناسازگاری بر اثر مقادیر نادرستی است که برای داده‌ها، به‌دلیل تداخل اجرای تراکنش‌ها به وجود می‌آید. الگوریتم‌های کنترل هم‌روندی، برای تضمین اجرای هم‌روند چندین تراکنش طراحی شده‌اند که به‌صورت هم‌روند با داده‌های مشترک کار می‌کنند. در این مقاله الگوریتم‌های کنترل هم‌روندی منتظرگذاشتن-میراندن (WD)[۴] و زخمی‌کردن-منتظرگذاشتن (WW)[۵] مدل‌سازی شده‌اند که جزء تکنیک‌های پیشگیری از بن‌بست هستند. از آنجا که شبکه پتری رنگی[۶] یکی از بهترین روش‌ها برای تحلیل مکانیزم‌های کنترل هم‌روندی است، مدل‌سازی‌ها با استفاده از پتری رنگی ارائه شده‌اند. پس از مدل‌سازی، به ارزیابی الگوریتم‌ها بر اساس پارامترهای تعداد تراکنش‌های واردشونده به سیستم، تعداد دستورهای هر تراکنش، تعداد داده‌های مشترک و غیرمشترک بین تراکنش‌ها و تعداد داده‌های مشترک در تراکنش‌هایی که هیچ دادهٔ غیرمشترکی ندارند، پرداخته شده است. پس از ارزیابی، این نتیجه به دست آمد که بر اساس پارامترهای ذکرشده، الگوریتم WW نسبت به WD زمان اجرای بسیار بهتری دارد.

۱- مقدمه

مکانیزم‌های کنترل هم‌روندی، برای حفظ انزوا در میان تراکنش‌های متعارض[۷] و حفظ سازگاری پایگاه داده‌ها استفاده می‌شوند [۳-۱]. مسئله کنترل هم‌روندی در پایگاه داده‌ها امری ضروری و بااهمیت است [۳]. در این زمینه تحقیقات فراوانی صورت گرفته است که نتیجه آن، به‌وجودآمدن الگوریتم‌های متنوع کنترل هم‌روندی است. با توجه به الگوریتم‌های متنوع در این زمینه و این واقعیت که روزبه‌روز بر اهمیت آن‌ها افزوده می‌شود، در حوزه ارزیابی الگوریتم‌های کنترل هم‌روندی جای کارِ بسیاری وجود دارد. مدل‌سازی صوری[۸] از الگوریتم‌های کنترل هم‌روندی در مطالعه ویژگی‌های مختلف آن‌ها بسیار مفید است [۲-۱]. بررسی‌ها نشان می‌دهد که شبکه پتری رنگی روشی مناسب برای مدل‌سازی مکانیزم‌های کنترل هم‌روندی است [۲-۱]. مکانیزم WD یکی از الگوریتم‌های پیشگیری از بن‌بست است که در آن حق تقدم زمانی تراکنش‌ها بر اساس زمان‌مهر و لحظه ورودشان به سیستم رعایت نمی‌شود؛ یعنی در مکانیزم WD هیچ قانونی وجود ندارد که تراکنشی که زودتر وارد سیستم شده، اولویت بیشتری برای دریافت سریع‌تر قفل‌های موردنیازش داشته باشد، به همین دلیل به آن الگوریتم نابازدارنده می‌گویند. در سمت مقابل، مکانیزم WW وجود دارد که یکی از الگوریتم‌های پیشگیری از بن‌بست است که در آن حق تقدم زمانی تراکنش‌ها بر اساس زمان‌مهر و لحظه ورودشان به سیستم رعایت می‌شود؛ یعنی در مکانیزم WW تراکنشی که زودتر وارد سیستم شده، اولویت بیشتری دارد که زودتر قفل‌های

*. پست الکترونیک نویسنده مسئول: saadatjou@sau.ac.ir
۱. استادیار، دانشکده مهندسی گروه کامپیوتر، دانشگاه علم و هنر یزد
۲. کارشناس ارشد، دانشکده مهندسی گروه کامپیوتر، دانشگاه علم و هنر
۳. دانشجوی دکتری، دانشکده مهندسی گروه کامپیوتر، دانشگاه کاشان

4. Wait – Die
5. Wound - Wait
6. Coloured Petri Net
7. Interfrence
8. Formal

موردنیازش را دریافت کند، به همین دلیل به آن الگوریتم بازدارنده می‌گویند.

در این مقاله ابتدا مکانیزم‌های WD و WW با استفاده از پتری رنگی مدل‌سازی شده‌اند. سپس الگوریتم‌ها با پارامترهای:

- تعداد تراکنش‌های واردشونده به سیستم [۴]
- سایز هر تراکنش (تعداد دستورهای هر تراکنش) [۵]
- تعداد داده‌های مشترک و غیرمشترک تراکنش‌ها [۶]
- تعداد داده‌های مشترک در تراکنش‌هایی بدون داده غیرمشترک [۶]

مقایسه و ارزیابی شده‌اند. آزمایش‌ها چندین بار تکرار و از مقادیر میانگین‌گیری شده است. نمودارهای لازم نیز برای مقایسه آسان‌تر ترسیم و بررسی شده‌اند.

مقاله به شکل زیر سازمان‌دهی شده است. در بخش دوم پیشینه تحقیق و مطالب مرتبط آورده و نحوه مدل‌سازی به‌طور کامل در بخش سوم بیان شده است. در بخش چهارم مدل‌ها بر اساس پارامترهای متفاوت با هم مقایسه شده‌اند. در بخش پنجم که بخش پایانی است، نتیجه‌گیری کلی از مباحث بیان‌شده مشاهده می‌شود.

۲- پیشینه تحقیق و مطالب مرتبط

در ابتدا بعضی از روش‌های پیاده‌سازی و مدل‌سازی موجود به‌همراه مزایا و معایب آن‌ها که تاکنون دیگران از آن‌ها استفاده کرده‌اند، بررسی می‌شوند. یکی از این روش‌ها شبکه پتری است. شبکه پتری، روشی صوری است که مزایای بسیاری از جمله آسان‌بودن و رابط کاربر گرافیکی بودن را داراست [۷]. به‌طور کلی شبکه‌های پتری ابزاری بسیار مفید برای مدل‌سازی و تحلیل عملیات هم‌روند، هم‌زمان، ناهم‌زمان، غیرقطعی، موازی یا توزیع‌شده دارند [۹-۸]. در [۱۲-۱۰] نیز مشاهده می‌شود که شبکه‌های پتری یکی از ابزارهای بسیار مفید برای شبیه‌سازی الگوریتم‌های کنترل هم‌روندی هستند و امکانات مختلف و موردنیاز برای شبیه‌سازی این الگوریتم‌ها را در اختیار ما قرار می‌دهند. در میان انواع مختلف شبکه‌های پتری، پتری رنگی قابلیت‌های مدل‌سازی زیادی دارد و یکی از بهترین روش‌ها برای تحلیل و تأیید مکانیزم‌های کنترل هم‌روندی است [۲-۱]. شایان ذکر است مدل‌کردن به‌وسیله پتری رنگی به‌راحتی می‌تواند برای تحلیل عملکرد نیز گسترش یابد.

در [۱۴-۱۳]، چند مورد از تکنیک‌های کنترل هم‌روندی پایگاه داده‌ها که به‌طور معمول استفاده می‌شوند، از طریق پیاده‌سازی در مقیاس کوچک، بررسی شده‌اند. در [۱۵] نیز، یک الگوریتم جدید برای کنترل هم‌روندی در پایگاه داده توزیع‌شده ارائه شده که از طریق پیاده‌سازی در مقیاس کوچک بررسی شده است. در [۱۶] یک تئوری از یک روش کنترل هم‌روندی جدید برای پایگاه داده‌ها بیان شده و در آن به کمک شبکه پتری نشان داده شده است که پروتکل پیشنهادی امکان‌پذیر است و می‌تواند نیازهای تراکنش بلادرنگ را برطرف کند. از طریق نظریه شبکه پتری، درستی این موضوع و روش کنترل هم‌روندی پیشنهادی ثابت شده است.

در [۵]، پس از تعیین برخی مفروضات، بازدهی الگوریتم کنترل هم‌روندیِ مرتب‌سازی زمان‌مهر پایه‌ای با کمک مدل مارکوف، مدل‌سازی شده است. از آنجا که ماهیت الگوریتم‌های کنترل هم‌روندی پیچیدگی خاصی دارد، به‌سختی میتوان عملکرد آن‌ها را از طریق تحلیل ریاضی بررسی کرد و این کار نیاز به انتخاب مفروضاتی برای مدل‌کردن الگوریتم دارد. به همین دلیل عملاً پیداکردن راه حل فرم بسته آن به کمک مدل مارکوف غیرممکن است و به این دلیل برای رفع مشکل مذکور، از مثال عددی استفاده شده است [۵]. در آن مقاله الگوریتم کنترل هم‌روندیِ مرتب‌سازی زمان‌مهر پایه‌ای، بعد از مدل‌سازی توسط مدل مارکف، بر اساس پارامتر تعداد دستورهای تراکنش‌ها ارزیابی شده است.

در [۷] تجزیه و تحلیل یک الگوریتم قفل متمرکز صورت گرفته است. این الگوریتم که قبلاً با استفاده از تجزیه و تحلیل‌های صف بررسی شده و مجدداً با شبکه پتری بسط‌یافته موردتحلیل قرار گرفته است. هرچند نتایج به‌دست‌آمده در این مطالعه نزدیک به نتایج حاصل از الگوریتم تحلیل‌شده در روش صف است، ولی دشواری‌های روش صف در مدل‌سازی آن مشاهده نشده است.

جدول ۱- توضیحات مربوط به مجموعه‌های رنگی

RESOURCE	برای نشان‌دادن نام منابع تعریف شده است.
SEQUENCE	برای نشان‌دادن دنباله‌ای از شماره دستورالعمل‌های تراکنش (با شروع از عدد یک) استفاده می‌شود.

ACCESS	برای مدل‌کردن نوع عملیاتی استفاده شده است که هر دستورالعمل می‌خواهد روی منابع داشته باشد. عملیات شامل: "LX" برای درخواست قفل انحصاری (قفل نوشتن و خواندن)، "LS" برای درخواست قفل اشتراکی (قفل خواندن)، "R" برای درخواست خواندن، "W" برای درخواست نوشتن و "UL" برای درخواست بازکردن قفل یک منبع است.
LOCK	برای نمایش‌دادن نوع قفلی استفاده می‌شود که در حال حاضر بر روی یک منبع موجود است. انواع قفل‌ها شامل: "LX" یعنی وجود قفل انحصاری، "LS" یعنی وجود قفل اشتراکی و "F" به‌معنی آزادبودن منبع و نبودن قفل است.
ثابت TransactionsNo	تعداد تراکنش‌های مدل را تعیین می‌کند. در تعریف مجموعه رنگ TRANSACTION استفاده شده است.
TRANSACTION	برای مدل‌کردن شناسه تراکنش‌ها استفاده می‌شود. این مجموعه رنگ شامل سه رنگ است: Trans(1)، Trans(2) و Trans(3) که سه تا از تراکنش‌های مدل را مشخص می‌کند.
BOOLEAN	از نوع بولی (بولین) تعریف شده است.
TIMESTAMP	برای نشان‌دادن زمان‌مهر هر تراکنش است.
MODE	وضعیت هر تراکنش را نشان می‌دهد. "normal" یعنی تراکنش در وضعیت عادی قرار دارد، "wait" یعنی در انتظار است و "can not commite" یعنی به‌دلیل وجود خطا، قادر به ادامه کار نیست؛ همه تراکنش‌ها باید طرد شوند و مشکل تراکنش رفع شود.
TRANSxULPO	برای مدل‌کردن شناسه یک تراکنش و وضعیت نقطه قفل آن تعریف شده است.
SEQxACCESS xRES	برای مدل کردن دستورالعمل های هر تراکنش استفاده شده است. لازم به ذکر است که هر دستورالعمل شامل شماره دستورالعمل، نوع عملیات و نام منبع می‌باشد.
REQUEST	برای هر تراکنش یک شماره درخواست را نگه می‌دارد. زمانی که مدل در وضعیت "normal" باشد، شماره درخواست مقدار صفر را خواهد داشت. اما زمانی که مدل در وضعیت "wait" باشد شماره درخواست، عددی غیر از صفر خواهد بود. تراکنشی که دارای شماره درخواست کمتری نسبت به بقیه باشد زودتر از سایرین انجام می‌شود و تراکنشی که دارای شماره درخواست بیشتری نسبت به بقیه باشد دیرتر از بقیه انجام می‌شود.
TRANSLIST	برای مدل کردن لیستی از تراکنش‌ها تعریف شده است. این لیست زمانی استفاده خواهد شد که چند مورد از تراکنش‌ها قفل اشتراکی بر روی یک منبع دارند.
TRANSLISTx LOCKxRES	برای مدل کردن قفل های موجود بر روی یک منبع است و لیست تراکنش‌هایی که بر روی یک منبع قفل دارند و نوع قفلشان را تعریف می‌کند.
Abort1xAbort2 xAbort3	برای تعین اینکه کدام تراکنش‌ها باید طرد شوند. هر تراکنشی که باید طرد شود قسمت Abort مربوط به آن دارای مقدار true می‌شود.
PRIOxxx	این مجموعه رنگ در درون یک توکن، اطلاعاتی برای هر سه تراکنش نگه می‌دارد. این اطلاعات شامل شماره دستور بعدی، زمان‌مهر هر سه تراکنش، و ضعیت آن‌ها و شماره درخواستشان است. بعلاوه دربرگیرنده این است که کدام تراکنش‌ها می‌توانند در حال حاضر انجام شوند و اینکه از بین تراکنش هایی که قادر به انجام هستند، کدام اولویت بیشتری دارد.

مدل‌سازی پروتکل تثبیت دو مرحله‌ای برای مدیریت تراکنش در محیط توزیع‌شده با استفاده از شبکه پتری زمانی، مطالعه شده است [۱۷]. خصوصیات و مشخصات صوری برای کنترل هم‌روندی تراکنش‌های پایگاه داده با استفاده از شبکه پتری معمولی بر اساس 2PL[1] نیز مورد مطالعه قرار گرفته است [۱۸] و این اطمینان را می‌دهد که توالی‌پذیری هم‌روندی زمان‌بندها بررسی شده است. مطالعه بازده کمی 2PL در سیستم‌های پایگاه داده موازی با

1. Two Phase Locking

استفاده از یک روش و متدولوژی و به کمک یک شبیه‌ساز دیگر در [۱۹] انجام شده است. روش‌های کپی اصلی و اولیه با استفاده از پتری سطح بالا که همان شبکه پتری رنگی است [۲۰] مدل شده‌اند. در [۲-۱] نیز یک مدل جدید از کنترل هم‌روندی برای پایگاه داده توزیع‌شده، بر اساس پروتکل 2PL با استفاده از پتری رنگی مدل شده است.

در [۴] یک الگوریتم امن و یک الگوریتم غیرامن برای پایگاه داده‌های بلادرنگ بعد از پیاده‌سازی در مقیاس کوچک، بر اساس پارامتر تعداد تراکنش‌های واردشونده به سیستم مقایسه شده‌اند. در [۶] یک مکانیزم بر اساس قفل دومرحله‌ای، بعد از پیاده‌سازی در مقیاس کوچک بر اساس پارامترهای تعداد داده‌های مشترک و غیرمشترک تراکنش‌ها و تعداد داده‌های مشترک در تراکنش‌هایی بدون داده غیرمشترک، ارزیابی شده است.

بعد از بررسی‌های صورت‌گرفته، مشاهده شد که هیچ‌یک از الگوریتم‌های WW و WD با استفاده از پتری رنگی مدل نشده‌اند و این الگوریتم‌ها بر اساس پارامترهای اشاره‌شده در بخش یک، ارزیابی نشده‌اند. به همین دلیل در این مقاله بعد از مدل‌سازی این الگوریتم‌ها، به ارزیابی آن‌ها پرداخته شده است.

۳- مدل‌سازی

از آنجا که نمایش گرافیکی شبکه‌های پتری، درک مسائل را ملموس‌تر می‌کند [۹-۷]، الگوریتم‌های WW و WD با استفاده از پتری رنگی و نرم‌افزار CPN Tools مدل شده‌اند.

اجزای مدل‌ها در شکل‌های (۱) و (۲) قابل‌مشاهده هستند. در این مدل‌سازی فرض شده است که هرکدام از تراکنش‌ها می‌توانند قسمتی از یک تراکنش بسیار بزرگ باشند. به این دلیل برای لحظه شروع به کار، قانونی وضع نشده است که تراکنشی که زمان مهر کمتری دارد، زودتر از بقیه شروع شود و تراکنشی که زمان مهر بیشتری دارد دیرتر از همه شروع بشود، ولی در صورت لزوم به راحتی می‌توان این تغییر را ایجاد کرد.

۳-۱- مجموعه‌های رنگ در مدل‌های WD و WW

مجموعه‌های رنگی این مدل‌ها برای سه تراکنش و دو منبع[1] در ادامه معرفی شده‌اند و توضیحات مربوط به آن‌ها در جدول ۱ آورده شده است.

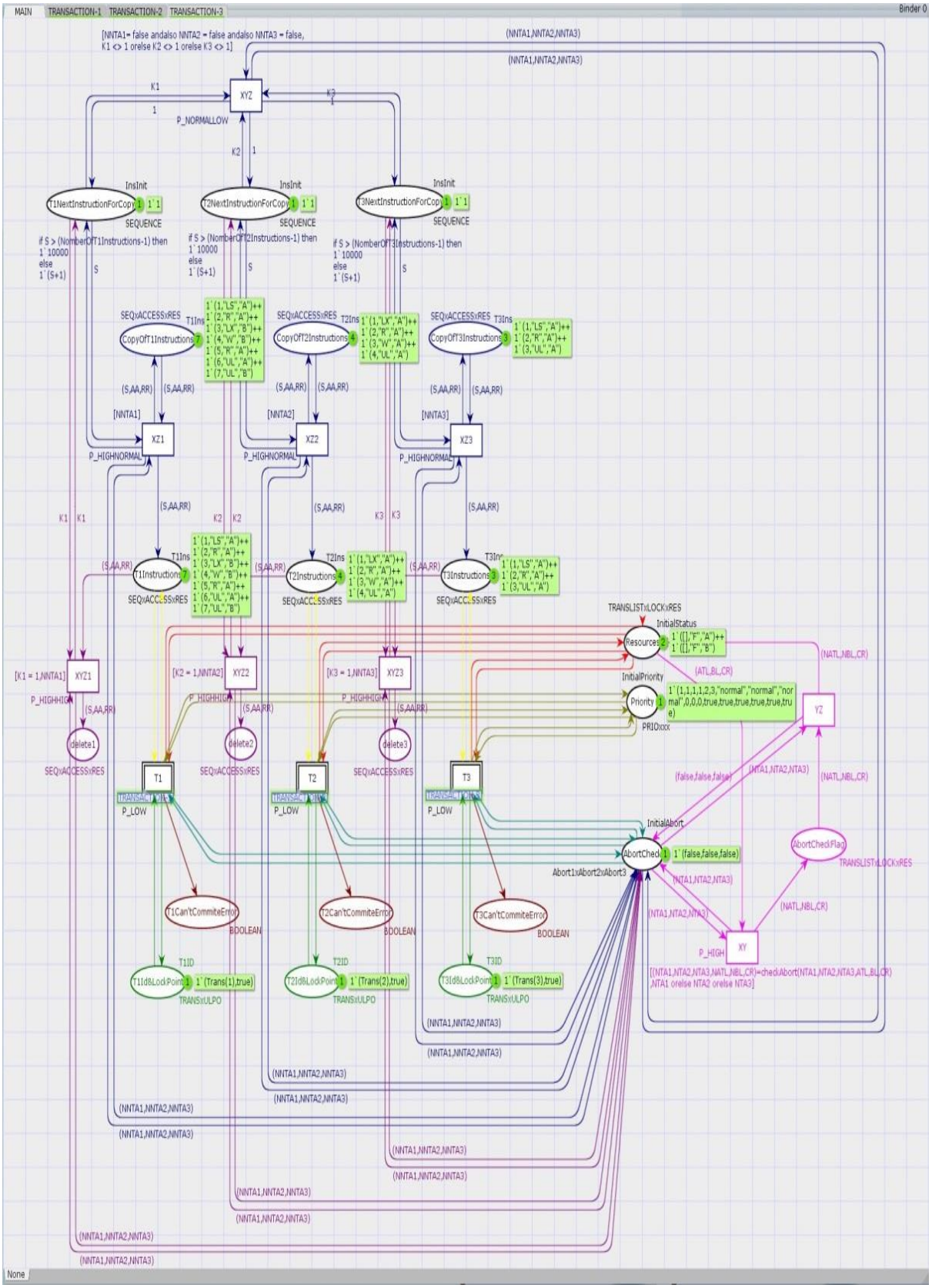
```
colset RESOURCE = string;
colset SEQUENCE = int;
colset ACCESS = string;
colset LOCK = string;
val TransactionsNO = 3;
colset TRANSACTION = index Trans with 1..
TransactionsNO;
colset BOOLEAN = bool;
colset TIMESTAMP = int;
colset MODE = string;
colset REQUEST = int;
closet TRANSxULP = product
TRANSACTION*BOOLEAN;
Closet SEQxACCESSxRES = product
SEQUENCE*ACCESS*RESOURCE;
colset TRANSLIST = list TRANSACTION;
colset TRANSLISTxLOCKxRES = product
TRANSLIST*LOCK*RESOURCE;
colset Abort1xAbort2xAbort3 = product
BOOLEAN*
BOOLEAN*BOOLEAN;
Colset PRIOxxx = product
SEQUENCE*SEQUENCE*
SEQUENCE*TIMESTAMP*TIMESTAMP*T
IMESTAMP*MODE*MODE*MODE*REQU
EST*REQUEST*REQUEST*BOOLEAN*B
OOLEAN*BOOLEAN*BOOLEAN*BOOLE
AN*BOOLEAN;
```

۳-۲- نشانه‌گذاری‌های اولیه در مدل‌های WD و WW

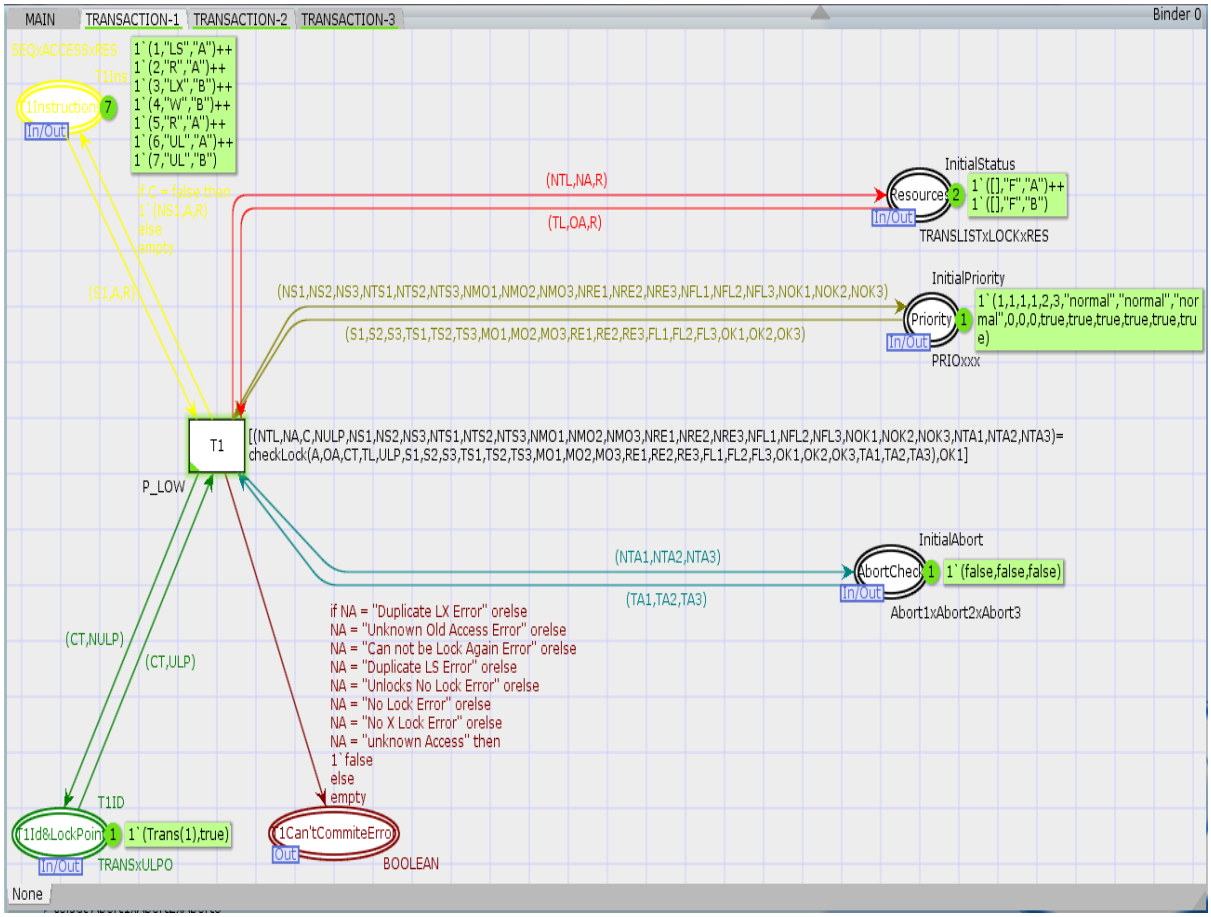
ثابت‌ها یا نشانه‌گذاری‌های اولیه مدل، به شرح زیر هستند که توضیحات مربوط به آن‌ها در جدول ۲ آمده است.

```
val InsInit = 1`1;
val ETL = [] : TRANSLIST;
val InitialStatus =
1`(ETL,“F”,“A”)++1`([],“F”, “B”);
val T1Ins = 1`(1,“LS”,“A”)++1`(2,“R”,“A”)++
1`(3,“LX”,“B”)++1`(4,“W”,“B”)++1`(5,“R”,“
A”)++1`(6,“UL”,“A”)++1`(7,“UL”,“B”);
val T1ID = 1`(Trans(1),true);
val InitialPriority =
1`(1,1,1,1,2,3,"normal","normal",
"normal",0,0,0,true,true,true,true,true,true);
val InitialAbort = 1`(false, false, false);
val NomberOfT1Instructions = 7;
```

1. Resource



شکل ۱: ماژول صفحه اصلی مدل‌های WD و WW به‌صورت سلسله‌مراتبی برای سه تراکنش



شکل ۲: ماژول مربوط به تراکنش T1 از مدل‌های WD و WW به‌صورت سلسله‌مراتبی برای سه تراکنش

۳-۳- متغیرهای موجود در مدل‌های WD و WW

متغیرهایی که در مدل‌ها استفاده شده‌اند به شرح زیر هستند:

```
Var S1,S2,S3,NS1,NS2,NS3,S,K1,K2,K3:
SEQUENCE;
var A,AA: ACCESS;
var OA,NA,BL,NBL: LOCK;
var
C,NOK1,NOK2,NOK3,OK1,OK2,OK3,TA1,
TA2,
TA3,NTA1,NTA2,NTA3,NNTA1,NNTA2,N
NTA3,FL1,
FL2,FL3,NFL1,NFL2,NFL3,ULP,NULP:
BOOLEAN;
var R,CR,RR: RESOURCE;
var CT: TRANSACTION;
var TL,NTL,ATL,NATL: TRANSLIST;
var TS1,TS2,TS3,NTS1,NTS2,NTS3:
TIMESTAMP;
Var MO1,MO2,MO3,NMO1,NMO2,NMO3:
MODE;
Var RE1,RE2,RE3,NRE1,NRE2,NRE3:
REQUEST;
```

جدول ۲- توضیحات مربوط به نشانه‌گذاری‌های اولیه مدل

InsInit	برای هر تراکنش، شماره دستورالعمل بعدی را که باید در محل موردنیاز کپی کند، در خود نگه می‌دارد.
ETL	یک لیست خالی از تراکنش‌ها را نشان می‌دهد. ثابت ETL در تعریف ثابت InitalStatus استفاده شده است.
InitialStatus	منابع موجود در سیستم را تعریف می‌کند. نشان می‌دهد که هر دو منبع موجود در سیستم در شروع کار آزادند.
T1Ins ، T2Ins ، T3Ins	به‌ترتیب د ستورالعمل‌های تراکنش شماره یک، دو و سه را نمایش می‌دهند.
InitialAbort	نشانه‌گذاری اولیه برای وضعیت طرد تراکنش‌ها را مشخص می‌کند. این مقدار در شروع برای تمام تراکنش‌ها برابر false است.

T1ID ،T2ID ، T3ID	به‌ترتیب شناسه تراکنش یک، دو، سه و همچنین وضعیت نقطه قفل این تراکنش‌ها را نشان می‌دهند. وضعیت نقطه قفل اگر true باشد، یعنی هنوز هیچ قفلی با این تراکنش باز نشده و اگر false باشد، یعنی حداقل یک قفل باز شده است و امکان قفل مجدد وجود ندارد.
InitialPriority	برخی اطلاعات مربوط به هر سه تراکنش را درون یک توکن نگه می‌دارد؛ اطلاعات مربوط به اینکه برای هرکدام از تراکنش‌ها کدام شماره دستور باید اجرا شود و زمان‌مهرهای سه تراکنش را معین می‌کند. وضعیت هر سه تراکنش، شماره درخواست‌های هر سه تراکنش، اطلاعات مربوط به اینکه کدام تراکنش‌ها قادر به انجام هستند و اینکه کدام تراکنش‌ها احتمال انجام دارند را در خود نگه می‌دارد.
NomberOfT1Instructions، NomberOfT2Instructions، NomberOfT3Instructions	تعداد دستورهای تراکنش‌ها را در خود نگه می‌دارد.

۳-۴- شرح عملکرد توابع WD و WW

بسیاری از قابلیت‌ها و عملکردهای مدل شبکه پتری بر اساس توابع نوشته‌شده توسط کاربر به زبان تابعی ML[1] صورت می‌گیرد [۲۲-۲۱]. در این بخش، عملیات توابع مدل‌ها توصیف شده‌اند. عملیات توابع به شرح زیر هستند:

getTransIndex: یک شناسه تراکنش و یک لیست از تراکنش‌ها را می‌گیرد و شاخص موقعیت تراکنش در لیست را برمی‌گرداند. توجه شود که «شمارش از صفر» است. اگر لیست شامل این تراکنش نباشد، تابع مقدار «۱-» را به‌عنوان نتیجه برمی‌گرداند.

eliminateTrans: یک شناسه تراکنش با نام T و یک لیست از شناسه‌های تراکنش‌ها به نام L را می‌گیرد. اگر تراکنش در لیست تراکنش‌ها وجود داشت، پس از آن تابع لیستی را بازمی‌گرداند که با از بین بردن تراکنش T از لیست اولیه ایجاد می‌شود، در غیر این صورت، لیست بدون تغییرِ تراکنش‌ها را برمی‌گرداند.

isExists: یک شناسه تراکنش و یک لیست از شناسه‌های تراکنش‌ها را می‌گیرد. اگر پارامتر اول، در لیست پارامتر دوم وجود داشته باشد، این تابع مقدار true، در غیر این صورت false را بازمی‌گرداند.

MAXrequ1Select، MAXrequ1Select و MAXrequ3Select: این توابع به‌ترتیب وضعیت تراکنش شماره یک، دو، سه و شماره درخواست هر سه تراکنش را می‌گیرند. با توجه به وضعیت تراکنش موردنظر و شماره درخواست‌های همه تراکنش‌ها، شماره درخواستی را برای تراکنش موردنظر ایجاد می‌کنند که به حالت انتظار رفته است. اگر تراکنش، تازه به حالت انتظار رفته باشد، شماره درخواست جدید برای آن در نظر گرفته می‌شود؛ به‌طوری که به بزرگ‌ترین شماره درخواست، یک واحد اضافه می‌شود و شماره درخواست تراکنش موردنظر معین می‌شود. ولی اگر تراکنش قبلاً در حالت انتظار بوده است، دیگر برای آن شماره درخواست جدیدی ایجاد نخواهد شد و شماره درخواست آن همان شماره قبلی خواهد ماند.

MAXtespSelect: زمان‌مهر هر سه تراکنش را می‌گیرد. با توجه به زمان‌مهرهای تراکنش‌ها، یک زمان‌مهر جدید برای تراکنشی که طرد شده است ایجاد می‌کند؛ به‌طوری که به بزرگ‌ترین زمان‌مهر، یک واحد اضافه می‌شود و زمان‌مهر تراکنش مورد نظر معین می‌شود.

checklock: فرض می‌کنیم TS(Ti) زمان‌مهر تراکنش Ti باشد و TS(Tj) زمان‌مهر تراکنش Tj باشد. برای مدل WD تابع checklock بررسی می‌کند که اگر TS(Ti)<TS(Tj) باشد، Ti که خواهان قفل روی داده است، انتظار می‌کشد تا Tj به اتمام برسد، در غیر این صورت، Ti طرد می‌شود تا دیرتر با مقدار زمان‌مهر جدید دوباره وارد سیستم شود و شروع به کار کند. برای مدل WW تابع checklock بررسی می‌کند که اگر TS(Ti)<TS(Tj) باشد و Tj روی داده قفل داشته باشد، Tj طرد می‌شود؛ در واقع داده از Tj گرفته می‌شود و به Ti که قدیمی‌تر است داده می‌شود. اما اگر این شرط برقرار نباشد، Ti باید انتظار بکشد. همچنین وضعیت تراکنش‌ها برای انجام مراحل بعدی را نیز مشخص می‌کند.

checkAbort: وظیفه دارد که در صورت طردشدن هرکدام از تراکنش‌ها، نام آن‌ها را از لیست تراکنش‌هایی که بر روی منبع موردنظر قفل دارند، حذف کند (در نهایت با

1. Meta Language

تکرار این تابع، نام تراکنش‌های طردشده را از لیست تمام منابع موجود حذف خواهد کرد).

۳-۵- تعیین اولویت برای فایر شدن گذار موردنظر از بین گذارهای فعال

مقادیر بیان‌شده در زیر برای تعیین اولویت اجرای گذار موردنظر از بین گذارهایی است که قادر به فایرشدن هستند. از بین گذارهایی که قادر به انجام‌اند، گذاری که دارای اولویت P_HIGHHIGH است، بالاترین اولویت را دارد و زودتر از بقیه اجرا خواهد شد و گذاری که دارای اولویت P_LOW است، پایین‌ترین اولویت را دارد و دیرتر از بقیه اجرا خواهد شد. سایر اولویت‌ها نیز بر اساس مقدارشان بین دو حالت ذکرشده قرار خواهند گرفت.

```
val P_HIGHHIGH = 10;
val P_HIGHNORMAL = 50;
val P_HIGH = 100;
val P_NORMAL = 1000;
val P_NORMALLOW = 5000;
val P_LOW = 10000;
```

۴- ارزیابی

در این بخش ارزیابی الگوریتم‌های WW و WD با استفاده از پارامترهای بیان‌شده در بخش اول، انجام خواهد شد. برای مشاهده نتایج دقیق‌تر، تمام آزمایش‌ها بیست مرتبه تکرار شده‌اند و از مقادیر، میانگین‌گیری به عمل آمده است.

برای ارزیابی‌های بخش ۴-۱ و همچنین برای اولین مجموعه دستورها در بخش ۴-۲، از مجموعه دستورهای موردآزمایش در [۲-۱] استفاده شده است. برای دومین مجموعه دستورها در بخش ۴-۲ و همین‌طور ارزیابی‌های بخش ۴-۳ و ۴-۴ روند مربوط به دستورهای تراکنش‌های [۲-۱] حفظ شده است و بر اساس آن تراکنش‌های بزرگ‌تر ایجاد شده‌اند؛ یعنی بعد از قفل اشتراکی "LS"، عملیات خواندن صورت گیرد و بعد از قفل انحصاری "LX"، عملیات نوشتن و خواندن. به این ترتیب دستورها گسترش یافته و تراکنش‌های بزرگ‌تر برای ارزیابی ارائه شده‌اند.

دلیل استفاده از این تراکنش‌ها، داشتن انواع عملیات شامل "LX"، "LS"، "R"، "W" و "UL" است(در جدول ۱ نیز به آن‌ها اشاره شده است). همچنین ذکر این نکته ضروری است که به‌دلیل نوع چیدمان دستورها امکان رخدادن انواع حالت‌ها از جمله "normal"، "wait" و "can not commite" (در جدول ۱ نیز به آن‌ها اشاره شده است) در آن‌ها وجود دارد.

۴-۱- پارامتر تعداد تراکنش‌های واردشونده به سیستم

در هر دو مدل WW و WD بررسی برای دو و سه تراکنش انجام شده است.

جدول ۳، تعداد مرحله‌های اجرای مدل WW و WD برای دو و سه تراکنش را نشان می‌دهد.

با توجه به مقادیر به‌دست‌آمده در جدول ۳، مشاهده می‌شود که در مدل WW و WD، هنگامی که تعداد تراکنش‌ها از دو به سه افزایش می‌یابد، سرعت اجرا کمی کمتر می‌شود. زمانی که دو تراکنش اجرا می‌شود، در مدل WW به‌طور متوسط هر دستور ۱,۵۳ مرحله و در مدل WD به طور متوسط هر دستور ۱,۴۷۲۶ مرحله، زمان برای اجرا صرف خواهد کرد. اما زمانی که سه تراکنش اجرا می‌شود، در مدل WW به طور متوسط هر دستور ۱,۸۰۹ مرحله و در مدل WD به طور متوسط هر دستور ۱,۸۹۹۳ مرحله، زمان برای اجرا صرف خواهد کرد. به این ترتیب با افزایش تعداد تراکنش‌ها، زمان اجرای هر دستور کمی افزایش می‌یابد.

در شکل (۳)، تعداد مرحله‌های اجرای دو و سه تراکنش در مدل‌های WW و WD مقایسه شده‌اند.

همانطور که در شکل (۳)، مشاهده می‌شود، در هنگامی که دو و سه تراکنش ورودی داریم، WW از نظر زمان اجرا نسبت به WD سرعت بسیار بیشتری دارد.

۴-۲- پارامتر تعداد دستورها هر تراکنش

در ابتدا زمان اجرای مدل‌ها با سه تراکنشی که دارای ۷، ۴ و ۳ دستور بودند محاسبه شد (این مجموعه از تراکنش‌ها، تراکنش‌های کوچک نامیده شده‌اند). سپس زمان اجرا برای سه تراکنشی که دارای ۷۰، ۴۰ و ۳۰ دستور بودند نیز اندازه‌گیری گردید (این مجموعه از تراکنش‌ها، تراکنش‌های بزرگ نامیده شده‌اند).

جدول ۴، تعداد مرحله‌های اجرای مدل‌های WW و WD برای تراکنش‌های کوچک و بزرگ را نشان می‌دهد.

با توجه به مقادیر به دست آمده در جدول ۴، مشاهده می‌شود که در مدل WW و WD، با افزایش تعداد دستورها تراکنش‌ها، برای اجرای هر دستور به زمان بیشتری نیاز خواهد بود. اما این موضوع امری عادی است، مخصوصاً هنگامی که تراکنش‌ها مانند تراکنش‌های استفاده شده در این آزمایش بوده و با منبع مشترک در ارتباط باشند. در

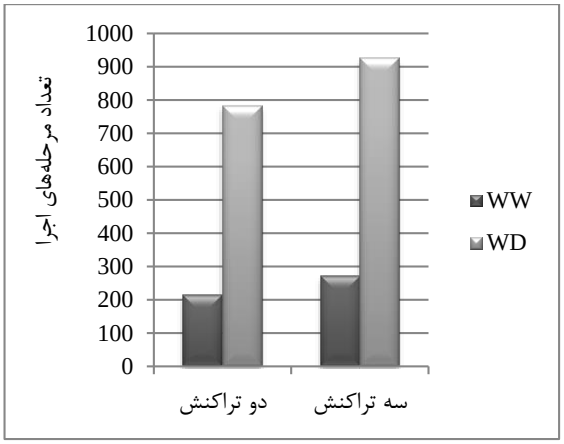
این حالت به دلیل طرد شدن‌های زیاد تراکنش‌ها، افزایش زمان امری طبیعی است. در مثال تراکنش‌های کوچک، در مدل WW به طور متوسط ۱,۷۰۳۵ مرحله و در مدل WD به طور متوسط ۴,۶۲۱۴ مرحله، زمان برای اجرای هر دستور صرف شد. اما در مثال تراکنش‌های بزرگ، در مدل WW به طور متوسط ۷,۱۸۱۴ مرحله و در مدل WD به‌طور متوسط ۷۷,۷۵۰۷ مرحله، زمان برای اجرای هر دستور صرف شد.

جدول ۳- تعداد گام‌های اجرای دو و سه تراکنش در مدل‌های WW و WD

تعداد تراکنش‌ها	WW		WD	
	دو	سه	دو	سه
تعداد مرحله‌ها	۱۵۲	۳۵۷	۱۵۲	۱۸۵۰
	۱۵۲	۱۵۶	۱۵۲	۲۳۸۰
	۱۵۲	۲۶۱	۲۹۵۸	۱۹۵۶
	۳۰۷	۴۷۱	۱۵۲	۱۵۶
	۱۵۲	۲۶۲	۱۵۲	۱۵۶
	۱۵۲	۲۶۲	۱۵۲	۱۵۶
	۱۵۲	۴۷۱	۱۵۲	۱۲۱۴
	۳۰۷	۱۵۶	۱۵۲	۱۵۶
	۱۵۲	۳۶۶	۱۵۲	۱۵۶
	۳۰۷	۲۶۲	۳۷۳۸	۱۵۶
	۳۰۷	۲۶۱	۱۵۲	۱۵۶
	۱۵۲	۲۶۲	۱۵۲	۳۰۲۶
	۱۵۲	۱۵۶	۱۵۲	۱۵۶
	۳۰۷	۱۵۶	۳۱۱۴	۱۵۶
	۱۵۲	۲۶۲	۳۴۲۶	۲۱۶۸
	۱۵۲	۳۶۶	۱۵۲	۱۴۲۶
	۳۰۷	۲۶۱	۱۵۲	۱۶۳۸
	۳۰۷	۲۶۱	۱۵۲	۱۵۶
	۳۰۷	۱۵۶	۱۵۲	۱۲۱۴
	۱۵۲	۲۶۲	۱۵۲	۱۹۵۶
میانگین	۲۱۴	۲۷۱,۳۵	۷۸۳,۴	۹۲۶,۹

در شکل (۴)، تعداد مرحله‌های اجرای سه تراکنش کوچک و در شکل (۵)، تعداد مرحله‌های اجرای سه تراکنش بزرگ، در مدل‌های WW و WD مقایسه شده‌اند.

همان‌طور که در شکل‌های (۴) و (۵)، مشاهده می‌شود، در مورد مثال تراکنش‌های کوچک و بزرگ، WW نسبت به WD دارای عملکرد بسیار بهتری است.



شکل ۳: مقایسه تعداد گام‌های اجرای دو و سه تراکنش در مدل‌های WW و WD

۴-۳- پارامتر تعداد داده‌های مشترک و غیرمشترک تراکنش‌ها

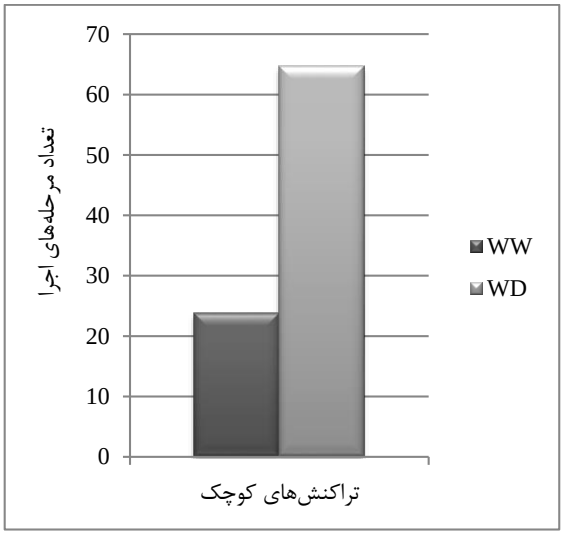
در این آزمایش، پارامتر موردبررسی تعداد داده‌های مشترک و غیرمشترک تراکنش‌ها است. شایان ذکر است که باید برای این آزمایش تعداد کلی داده‌ها ثابت در نظر گرفته شود. تعداد داده‌های مشترک و غیرمشترک به‌نوعی با هم مرتبط و مکمل یکدیگرند. کاهش داده‌های غیرمشترک به معنی افزایش داده‌های مشترک و برعکس، افزایش داده‌های غیرمشترک به‌معنی کاهش داده‌های مشترک است.

در ابتدا زمان اجرای مدل‌ها با سه تراکنشی که دارای یک داده غیرمشترک و ۱۴ داده مشترک بودند محاسبه شد (این مجموعه از تراکنش‌ها، تراکنش‌ها با داده‌های غیرمشترک کم، نامیده شده‌اند). سپس زمان اجرا برای سه تراکنشی که دارای ۱۰ داده غیرمشترک و ۵ داده مشترک بودند نیز اندازه‌گیری شد (این مجموعه از تراکنش‌ها، تراکنش‌ها با داده‌های غیرمشترک زیاد، نامیده شده‌اند).

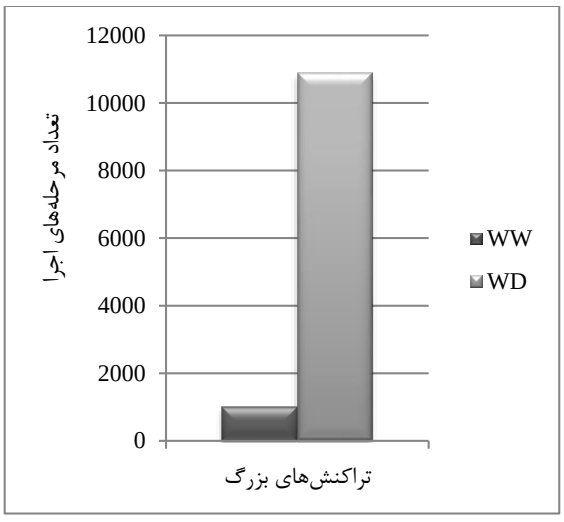
جدول ۵، تعداد مرحله‌های اجرای مدل‌های WW و WD برای تراکنش‌ها، با تعداد کم و زیادِ داده‌های غیرمشترک را نشان می‌دهد.

با توجه به مقادیر به‌دست‌آمده در جدول ۵، مشاهده می‌شود که در مدل WW و WD، تعداد داده‌های غیرمشترک و مشترک تراکنش‌ها در زمان اجرای تراکنش‌ها مؤثر است. هنگامی که تعداد داده‌های غیرمشترک تراکنش‌ها کم است و تعداد داده‌های مشترک زیاد، در مدل WW به‌طور متوسط هر دستور ۱,۶۵۴۸ مرحله و در مدل WD به‌طور متوسط هر دستور ۲۰,۸۴۳۲ مرحله، زمان برای اجرا صرف خواهد کرد. اما زمانی که تعداد داده‌های غیرمشترک

تراکنش‌ها زیاد است و تعداد داده‌های مشترک کم، در مدل WW به‌طور متوسط هر دستور ۱,۵۹۳۹ مرحله و در مدل WD به‌طور متوسط هر دستور ۱۸,۵۳۹۴ مرحله، زمان برای اجرا صرف خواهد کرد.



شکل ۴: مقایسه تعداد گام‌های اجرای تراکنش‌های کوچک در مدل‌های WW و WD



شکل ۵: مقایسه تعداد گام‌های اجرای تراکنش‌های بزرگ در مدل‌های WW و WD

همان‌طور که در شکل (۶)، مشاهده می‌شود، در هر دو حالت (تراکنش‌ها با تعداد کم و زیادِ داده‌های غیرمشترک) WW از نظر زمان اجرا نسبت به WD سرعت بسیار بیشتری دارد. در WW و WD هرچه تعداد داده‌های غیرمشترک کمتر و تعداد داده‌های مشترک بیشتر باشد، زمان اجرا نیز کمتر خواهد بود و هرچه تعداد داده‌های غیرمشترک زیادتر و تعداد داده‌های مشترک کمتر شود، زمان اجرا نیز زیادتر خواهد شد. شایان ذکر است که WD نسبت به تعداد داده‌های غیرمشترک بسیار حساس است و با افزایش تعداد داده‌های غیرمشترک و کاهش تعداد داده‌های مشترک، زمان اجرا نیز به میزان چشم‌گیری افزایش می‌یابد.

جدول ۴- تعداد گام‌های اجرای تراکنش‌های کوچک و بزرگ در مدل WW و WD

تراکنش ها	WW		WD	
	تراکنش های کوچک	تراکنش‌های بزرگ	تراکنش‌های کوچک	تراکنش‌های بزرگ
تعداد مرحله ها	۲۹	۹۵۳	۵۳	۱۰۴۶۰
	۲۵	۱۰۲۲	۸۴	۹۵۱۸
	۲۶	۱۱۰۷	۹۸	۱۱۰۰۲
	۲۸	۱۰۲۳	۱۴	۹۲۸۴
	۲۷	۹۵۶	۱۵	۹۲۹۵
	۲۷	۹۵۲	۷۹	۱۱۴۱۵
	۲۷	۱۰۲۱	۱۴	۱۱۱۵۶
	۲۵	۱۰۴۲	۷۰	۱۰۳۴۰
	۱۶	۹۵۷	۱۶۸	۹۵۷۹
	۴۰	۱۱۶۶	۱۴	۹۹۵۹
	۲۷	۹۵۸	۱۸۲	۱۱۵۴۷
	۱۵	۱۰۲۱	۹۲	۱۱۳۰۴
	۱۶	۱۰۴۵	۹۳	۱۱۹۶۷
	۱۵	۸۹۰	۳۰	۱۴۹۶۶
	۳۰	۹۵۶	۶۴	۹۷۶۹
	۲۹	۱۰۲۵	۴۲	۱۰۵۰۶
	۲۶	۱۰۴۴	۱۴	۱۲۰۴۸
	۲۹	۹۷۵	۷۱	۱۰۷۳۳
	۲۵	۱۰۳۸	۵۷	۱۱۷۴۵
	۱۵	۹۵۷	۴۰	۱۱۱۰۹
میانگین	۲۳,۸۵	۱۰۰۵,۴	۶۴,۷	۱۰۸۸۵,۱

در شکل (۶)، تعداد مرحله‌های اجرای سه تراکنش با تعداد کمِ داده‌های غیرمشترک و سه تراکنش با تعداد زیادِ

داده‌های غیرمشترک در مدل‌های WW و WD مقایسه شده‌اند.

جدول ۵- تعداد گام‌های اجرای تراکنش‌ها با تعداد کم و زیاد داده‌های غیرمشترک در مدل WW و WD

تراکنش-ها	WW		WD	
	داده‌های غیرمشترک کم	داده‌های غیرمشترک زیاد	داده‌های غیرمشترک کم	داده‌های غیرمشترک زیاد
تعداد مرحله‌ها	۳۱۵	۱۷۵	۳۵۱۶	۱۹۲۰
	۱۷۵	۱۷۵	۲۸۲۶	۲۴۴۸
	۲۵۴	۱۷۶	۱۹۸۰	۸۲۵۲
	۳۱۵	۲۵۶	۳۶۷۰	۵۰۹۸
	۴۹۰	۲۶۳	۲۵۵۱	۴۲۱۵
	۱۷۵	۲۶۳	۵۱۹۶	۳۵۱۰
	۳۱۴	۲۶۱	۱۷۵	۷۷۲۴
	۱۷۶	۱۷۵	۳۸۱۱	۱۷۵
	۳۱۰	۳۴۹	۳۱۰۷	۶۳۱۶
	۱۷۵	۲۵۶	۱۹۸۱	۴۲۱۹
	۳۱۶	۱۷۵	۱۴۲۰	۱۷۵
	۲۱۳	۲۶۱	۳۶۶۷	۲۰۹۶
	۳۱۵	۳۵۰	۴۳۵۶	۲۴۴۸
	۱۷۶	۱۷۵	۴۷۸۹	۱۳۹۴
	۱۷۵	۲۶۳	۲۹۷۰	۲۰۹۶
	۳۱۶	۲۵۶	۳۵۲۶	۱۷۶
	۱۷۵	۱۷۵	۴۲۲۵	۳۶۸۵
	۲۱۴	۲۶۱	۳۶۶۹	۱۷۵
	۳۱۶	۱۷۵	۲۲۶۰	۱۵۴۶
	۲۴۸	۱۷۷	۵۳۳۶	۱۷۵
میانگین	۲۵۸,۱۵	۲۴۸,۶۵	۳۲۵۱,۵۵	۲۸۹۲,۱۵

۴-۴- پارامتر تعداد داده‌های مشترک در تراکنش‌هایی بدون داده غیرمشترک

در این آزمایش پارامتر موردبررسی تعداد داده‌های مشترکِ بین تراکنش‌ها است، البته هیچ داده غیرمشترکی در بین این تراکنش‌ها وجود ندارد. در ابتدا زمان اجرای مدل‌ها، با سه تراکنشی که دارای ۲ نوع داده مشترک بودند محاسبه شد (این مجموعه از تراکنش‌ها، تراکنش‌ها با داده‌های مشترک کم و بدون داده غیرمشترک، نامیده شده‌اند). سپس زمان اجرا با سه تراکنشی که دارای ۲۰ نوع داده مشترک بودند نیز اندازه‌گیری شد (این مجموعه از تراکنش‌ها، تراکنش‌ها با داده‌های مشترک زیاد و بدون داده غیرمشترک، نامیده شده‌اند).

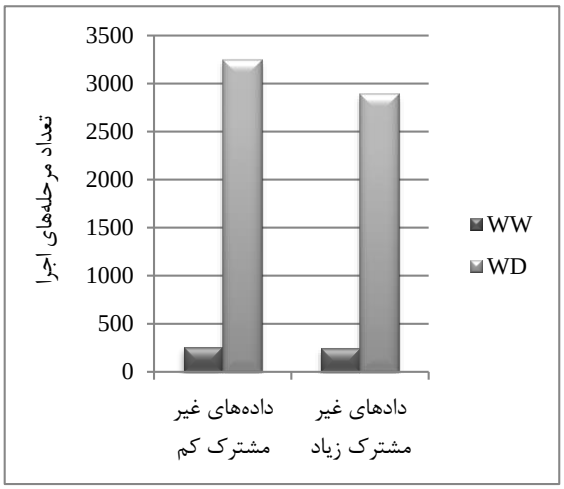
جدول ۶- تعداد گام‌های اجرای تراکنش‌هایی بدون داده غیرمشترک، با تعداد کم و زیاد داده‌های مشترک در مدل WW و WD

تراکنش‌ها	WW		WD	
	داده‌های مشترک کم و بدون داده غیرمشترک	داده‌های مشترک زیاد و بدون داده غیرمشترک	داده‌های مشترک کم و بدون داده غیرمشترک	داده‌های مشترک زیاد و بدون داده غیرمشترک
تعداد مرحله‌ها	۳۱۹	۲۰۸	۱۰۶۴۸	۸۹۴۶
	۴۴۳	۳۶۲	۷۸۱۰	۱۱۸۹
	۳۱۸	۲۱۵	۱۳۶۷۲	۱۸۹۸۲
	۴۴۵	۳۶۹	۹۰۷۸	۱۹۹۷۱
	۳۲۱	۲۱۱	۱۳۴۲۰	۲۱۶
	۴۴۶	۴۹۵	۱۳۹۲۴	۶۶۲۹
	۳۲۱	۵۳۲	۷۲۴۷	۱۶۰۸۰
	۳۲۰	۳۶۷	۱۵۴۳۶	۱۳۵۲
	۴۴۶	۲۱۵	۹۹۶۳	۱۷۰۳۸
	۴۴۶	۲۱۳	۷۶۸۲	۲۱۵
	۳۲۱	۳۶۸	۸۶۳۳	۲۱۶
	۳۲۱	۳۷۲	۸۱۹۲	۱۶۲۲۸
	۴۴۶	۳۷۴	۱۴۵۵۴	۱۳۵۲
	۴۴۵	۳۶۸	۱۱۹۰۸	۱۰۴۰۷
	۳۲۰	۳۶۰	۱۲۱۶۰	۹۱۰۹
	۳۲۱	۴۹۰	۱۴۱۷۶	۱۶۸۷۶
	۴۴۴	۲۱۵	۷۹۳۵	۱۸۴۹۶
	۳۱۹	۴۹۴	۱۲۲۸۶	۲۱۵
	۳۱۸	۴۸۸	۶۴۱۲	۲۱۴
	۳۲۰	۳۶۳	۹۲۰۷	۱۸۰۲۰
میانگین	۳۷۰	۳۵۳,۹۵	۱۰۷۱۷,۱۵	۹۰۸۷,۵۵

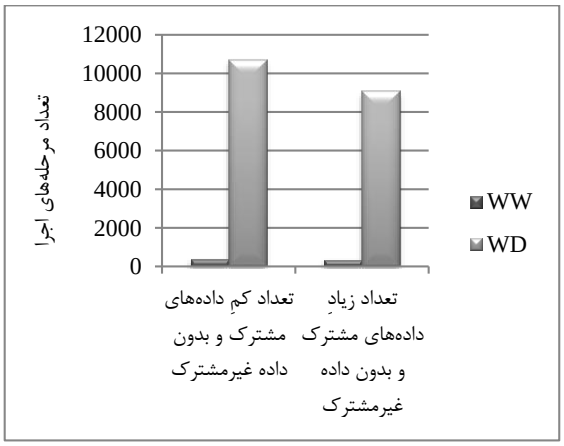
جدول ۶، تعداد مرحله‌های اجرای مدل‌های WW و WD برای تراکنش‌ها، با تعداد کم و زیاد داده‌های مشترک و بدون داده غیرمشترک را نشان می‌دهد.

با توجه به مقادیر به‌دست‌آمده در جدول ۶، مشاهده می‌شود که در مدل WW و WD، تعداد داده‌های مشترک در تراکنش‌هایی که داده غیرمشترک ندارند، بر روی زمان اجرای تراکنش‌ها تأثیر می‌گذارد. هنگامی که تعداد داده‌های مشترک تراکنش‌هایی که داده غیرمشترک ندارند، کم است، در مدل WW به‌طور متوسط هر دستور ۱,۹۴۷۳ مرحله و در مدل WD به‌طور متوسط هر دستور ۵۶,۴۰۶۰ مرحله، زمان برای اجرا صرف خواهد کرد. اما زمانی که

تعداد داده‌های مشترک در تراکنش‌های بدون داده غیرمشترک، زیاد است، در مدل WW به‌طور متوسط هر دستور ۱,۸۶۲۸ مرحله و در مدل WD به‌طور متوسط هر دستور ۴۷,۸۲۹۲ مرحله، زمان برای اجرا صرف خواهد کرد. مشاهده شد زمانی که داده‌های مشترک در تراکنش‌هایی که داده غیرمشترک ندارند، زیاد می‌شود، سرعت اجرای دستورها کمی افزایش می‌یابد. این موضوع به این دلیل رخ می‌دهد که در مواقعی که مانند این آزمایش تعداد دستورها ثابت است، اگر دستورها فقط با تعداد اندکی داده مشترک در ارتباط باشند، باید زمان بیشتری در انتظار باشند یا طرد شوند تا کار دیگری تمام شود و بتوانند داده را در اختیار بگیرند. اما اگر دستورها با تعداد زیادی داده مشترک در ارتباط باشند باید زمان کمتری منتظر بمانند تا کار دیگری تمام شود و راحت‌تر می‌توانند داده را در اختیار بگیرند.



شکل ۶: مقایسه نعداد گام‌های اجرای تراکنش‌ها با تعداد کم و زیاد داده‌های غیرمشترک در مدل‌های WW و WD



شکل ۷: مقایسه زمان اجرای تراکنش‌ها با تعداد کم و زیاد داده‌های مشترک و بدون داده غیرمشترک در مدل‌های WW و WD

در شکل (۷)، تعداد مرحله‌های اجرای سه تراکنش با تعداد داده‌های مشترک کم و بدون داده غیرمشترک، همچنین سه تراکنش با تعداد داده‌های مشترک زیاد و بدون داده غیرمشترک، در مدل‌های WW و WD مقایسه شده‌اند.

همان‌طور که در شکل (۷)، مشاهده می‌شود، در هر دو حالت (تراکنش‌ها بدون داده غیرمشترک، با تعداد کم و زیاد داده‌های مشترک)، عملکرد WD نسبت به WW در وضعیت بدتری قرار دارد و زمان اجرای آن از WW بیشتر است. همچنین لازم به ذکر است که WD نسبت به تعداد داده‌های مشترک (در تراکنش‌هایی که داده غیرمشترک ندارند) بسیار حساس است و با افزایش تعداد داده‌های مشترک، زمان اجرا نیز به میزان چشم‌گیری کاهش می‌یابد. در هر دو مدل، زمانی که تراکنش‌ها داده غیرمشترک ندارند، هرچه تعداد داده‌های مشترک کمتر باشد، زمان اجرا بیشتر و هرچه تعداد داده‌های مشترک زیادتر شود زمان اجرا کمتر خواهد شد.

۵- نتیجه‌گیری

با توجه به افزایش اهمیت هم‌روندی در سیستم مدیریت پایگاه داده‌ها و همچنین گسترش روزافزون انواع پایگاه داده‌ها در سراسر جهان، نیاز به تشخیص بهترین پروتکل‌های کنترل هم‌روندیِ پایگاه داده‌ها، بیشتر نمایان می‌شود. در این راستا الگوریتم‌های کنترل هم‌روندی WW و WD بررسی شده‌اند. برای ارزیابی پروتکل‌ها، اولین گام پیاده‌سازی یا مدل‌سازی آن‌ها بود. با بررسی‌های صورت‌گرفته، این نتیجه به دست آمد که شبکه‌های پتری رنگی روشی قدرتمند برای مدل‌سازی و تجزیه و تحلیل سیستم‌های هم‌روند هستند. در این مقاله مدل‌های WW و WD به‌صورت مدل‌های قابل‌گسترش با استفاده از شبکه‌های پتری رنگی ارائه شده‌اند.

بعد از مدل‌سازی به ارزیابی مدل‌ها بر اساس پارامترهای تعداد تراکنش‌های واردشونده به سیستم، تعداد دستورهای هر تراکنش، تعداد داده‌های مشترک و غیرمشترک بین تراکنش‌ها و تعداد داده‌های مشترک در تراکنش‌هایی بدون داده غیرمشترک، پرداخته شد.

با بررسی تک‌تک پارامترهای موردارزیابی، این نتیجه به دست آمد که به‌طور کلی الگوریتم WW نسبت به WD عملکرد بهتری دارد. الگوریتم WD از نظر زمان اجرا با اختلاف زیادی در سطح بدتری نسبت به الگوریتم WW قرار دارد.

۶- مراجع

[1] Pashazadeh, S., "Modeling and verification of deadlock potentials of a concurrency control mechanism in distributed databases using hierarchical colored petri net", International Journal of Information and Education Technology (IJIET), Vol. 2, No. 2, 2012, pp. 77-82.

[2] Pashazadeh, S., "Modeling a concurrency control Mechanism in distributed databases using hierarchical colored petri net", International Conference on Information and Computer Applications (ICICA), Singapore, Vol. 24, 2012, pp. 286-289.

[3] Shu, L. C., Young, M.,"Versioning concurrency control for hard real-time systems", Journal of Systems and Software, Vol. 63, No. 3, 2002, pp. 201-218.

[4] Hedayati, M., Kamali, S. H., Shakerian, R., Rahmani, M., "Evaluation of performance concurrency control algorithm for secure firm real-time database systems via simulation model", Networking and Information Technology (ICNIT), International Conference on IEEE, 2010, pp. 260-264.

[5] Singhal, M., "Performance analysis of the basic timestamp ordering algorithm via Markov modeling—performance evaluation", Performance Evaluation, Vol. 12, No. 1, 1991, pp. 17-41.

[6] Al-Jumah, N. B., Hossam, S. H., El-Sharkawi, M., "Implementation and modeling of two-phase locking concurrency control—a performance study", Information and Software Technology, Vol. 42, No. 4, 2000, pp. 257-273.

[7] Ozsu, M. T.,"Modeling and analysis of distributed database concurrency control algorithms using an extended petri net formalism, Software Engineering", Transactions on IEEE, Vol. SE-11, No. 10, 1985, pp. 1225-1240.

[8] Mikkilineni, K. P., Chow, Y. C., Su, S. Y. W., "Petri-net-based modeling and evaluation of pipelined processing of concurrent database queries", Software Engineering, Transactions on IEEE, Vol. 14, No. 11. 1988.

[9] Han, Y., Jiang, C., Luo, X., "A study of concurrency control in web-based distributed real-time database system using extended time petri nets, Parallel Architectures", Algorithms and Networks (ISPAN'04), in Proceedings of the 7th International Symposium on IEEE, 2004, pp. 67-72.

[10] Murata, T., "Petri nets: properties, analysis and applications", in Proceedings of IEEE, Vol. 77, No. 4, 1989, pp. 541-580.

[11] Devillers, R., Best, E., "Sequential and concurrent bihavior in petri net theory, Theorical computer science", Vol. 55, No. 1, 1987, pp. 87-136.

[12] Seatzu, C., Cabasino, M. P., Giua, A., "Introduction to petri nets", Control of Discrete-Event Systems (LNCIS), Vol. 433, 2013, pp. 191–211.

[13] Zhen, C., Li, K., "Improved distributed concurrency control algorithm based on real-time database systems", Computational Intelligence and Software Engineering, (CiSE), International Conference on IEEE, 2009, pp. 1-3.

[14] Lee, J., "Precise serialization for optimistic concurrency control", Data & Knowledge Engineering, Vol. 29, No. 2, 1999, pp. 163-178.

[15] Mousavi, S. M. A., Naji, H. R., Ebrahimi, A. R., "Optimization of majority protocol for controlling transactions concurrency in distributed databases by multi-agent systems", International Journal of Applied Operational Research, Vol. 3, No. 1, 2013, pp. 95-108.

[16] Chen, J., Wang, Y. F., Wang, J. P., "Concurrency control protocol for real-time database and the analysis base on petri net", Advanced Materials Research, Vols. 143-144, 2011, pp. 12-17.

[17] Sarkar, B. B., Nabendu, C., "Modeling & analysis of transaction management for distributed database environment using Petri Nets", In Nature & Biologically Inspired Computing (NaBIC), World Congress on IEEE, 2009, pp. 918-923.

[18] Jie, H., Fengying, L., Huijiao, W., "Petri net based model for concurrent control of database system", International Conference on Intelligent Computing and Integrated Systems (ICISS), 2010, pp. 813-815.

[19] Jenq, B-C., Twichell, B. C., Keller, T. W., "Locking performance in a shared nothing parallel database machine", Knowledge and Data Engineering, Transactions on IEEE, Vol. 1, No. 4, 1989, pp. 530-543.

[20] Voss, K., "Prototyping and verifying distributed database systems using executable high-level Petri net models, Systems", Man, and Cybernetics,. Computational Cybernetics and Simulation., International Conference on IEEE, Vol. 4, 1997, pp. 3395-3400.

[21] Paulson, L. C., "ML for the working programmer", (2th ed.), NY. USA, Press Cyndicate of the University of Cambridge, 1996.

[22] Harper, R., "Programming in standard ML, Pittsburgh United States", Carnegie Mellon University, 2001.