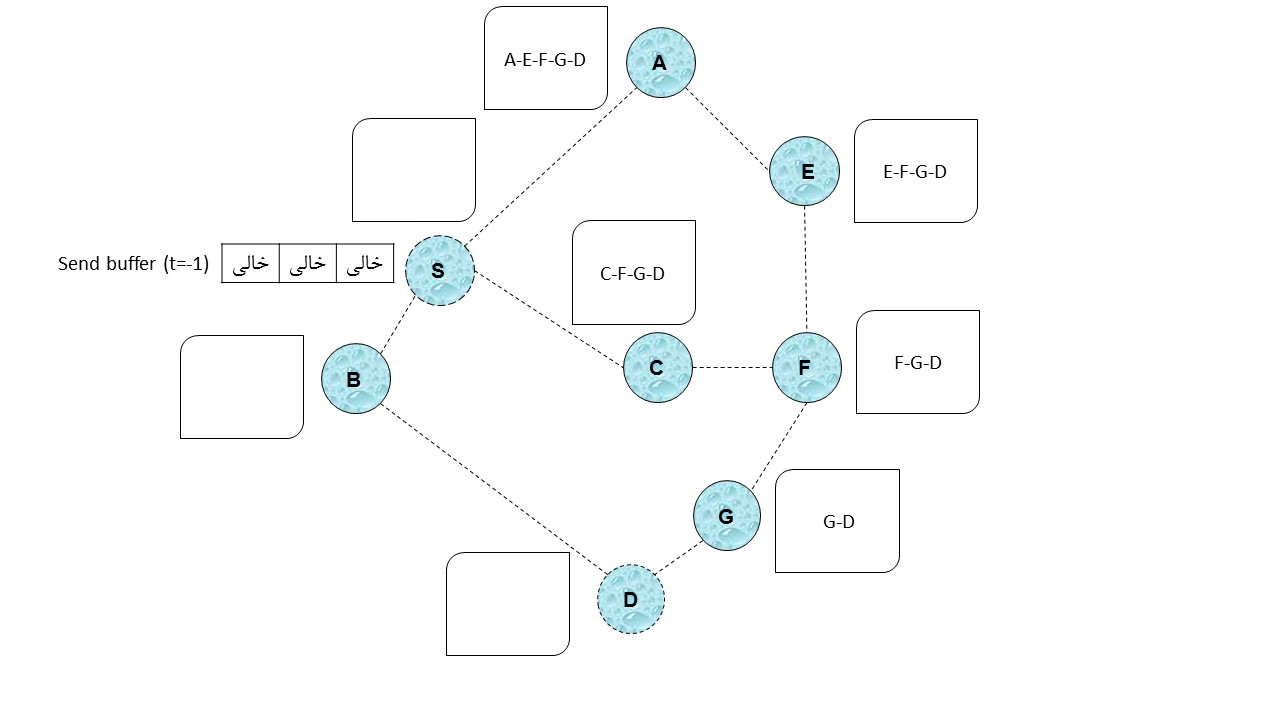
**کشف مسیر در DSR**

**سوال اول -** در شکل زیر، همبندی شبکه و حافظه نهان[[1]](#footnote-1) هر گره نمایش داده شده است. خطوط نقطه‌چین مشخص‌کننده پیوند دو طرفه بین دو گره منتهاالیه می‌باشد. فرض می‌کنیم مسیریابی در این شبکه با استفاده از پروتکل DSR انجام می‌گیرد و همه مکانیزم‌های الحاقی این پروتکل مسیریابی به استثنای محدودیت تعداد گام‌های درخواست مسیر[[2]](#footnote-2) فعال می‌باشد. به علاوه، فاصله زمانی دریافت یک بسته از گره همسایه[[3]](#footnote-3) 1 میلی‌ثانیه و زمان دریافت یک بسته ارسال شده از یک گره به یک گره دیگر وابسته به تعداد گام فاصله بین آن دو گره تعریف شده است، به طوریکه در صورت برابر بودن فاصله دو گره نسبت به یک گره مشخص، ترتیب دریافت بر اساس ترتیب الفبایی برچسب حرف انگلیسی معرف گره می‌باشد. لازم به ذكر است در شمارش تعداد گام بين دو گره خود گره مبدا شمارش نمي‌شود و زمان پردازش بسته‌ها در گره‌های میانی ناچیز در نظر گرفته شده است. در این شکل، گره مبدا S و مقصد D با خط‌چین مشخص شد‌ه‌اند و مبدا برای 10 میلی‌ثانیه ترافیک داده‌ای را با نرخ ثابت[[4]](#footnote-4) 1 به مقصد D تولید می‌کند و این ترافیک، تنها ترافیک تولید شده در گره S می‌باشد. بسته‌های داده به ترتیب تولید از 1 شماره‌گذاری شده‌اند. به علاوه، تعداد بسته‌های قابل ذخیره‌سازی در بافر ارسال[[5]](#footnote-5) هر گره 3 بسته می‌باشد و زمان‌سنج انقضای هر بسته در بافر ارسال به میزان دو برابر زمان لازم برای عبور از قطر شبکه تنظیم شده است. مکانسیم جایگزینی و اخراج بسته­ها در این بافر به صورت FIFO می­باشد. همچنین، مقدار شناسه درخواست یکتا[[6]](#footnote-6) در گره S برای اولین بسته درخواست مسیر[[7]](#footnote-7) (RREQ) برابر 100 می‌باشد و مقدار اولیه حداقل فاصله زمانی بین دو بسته RREQ به یک مقصد یکسان برابر 1 میلی‌ثانیه تنظیم شده است (این مقدار واقع‌گرایانه نیست و صرفا جهت تمرین آورده شده است). در ادامه سوالات، زمان کنون را t = 0 در نظر می‌گیریم و بقیه زمان‌های خواسته شده در سوالات را بر مبنای زمان سراسری نسبت به t = 0 می‌نویسیم.



**الف)** زمان‌سنج تاخیر زمانی ارسال بسته پاسخ مسیر[[8]](#footnote-8) (RREP) در چه گره‌هایی و در چه زمانی جهت پیشگیری از RREPهای سیل‌آسا[[9]](#footnote-9)، فعال می‌شود. زمان انقضای این زمان‌سنج را برای هر یک از گره‌های مورد نظر روی شکل، بنویسید. پارامتر(های) تصادفی مورد نیاز جهت تنظیم زمان‌سنج را ناچیز در نظر بگیرید.

**پاسخ:**

این اتفاق برای گره‌های همسایه‌ای که مسیر رسیدن به مقصد را در کش خود داشتند خواهد افتاد. یعنی گره‌های A, C.

**در t = 1 بسته‌ی RREQ به این گره‌ها می‌رسد و تایمر فعال می‌شود.**

زمانی که برای این تایمرها ست می‌شود از فرمول H (h – 1 + r) به دست می‌آید. r یک مقدار رندوم ناچیز است و صفر در نظر میگیریم. H باید ۲ برابر t\_propagation باشد پس برابر ۲ است (فاصله زمانی دریافت بسته‌ها ۱ است) و h هم برابر تعداد هاپ از مقصد است. پس داریم:

**TA = 2 \* (4 – 1 + 0) = 6**

**TC = 2 \* (3 – 1 + 0) = 4**

**ب)** اولین RREP توسط چه گره‌ای (گره مولد RREP) و در چه زمانی برگردانده می‌شود. موارد مدنظر خود جهت پاسخگویی به این سوال را شرح دهید.

**پاسخ:**

**اولین پاسخ از گره D ارسال می‌شود.**

بسته‌ی RREQ در t = 1 به B می‌رسد و چون در کش چیزی ندارد پس به سمت D میرود و در t = 2 به مقصد میرسد.

اینجا در t = 2 پاسخ به سمت مبدا ارسال می‌شه و در t = 4 به مبدا میرسه.

**پس ارسال RREP در t = 2 توسط D انجام میشه و در t = 4 به مبدا میرسه.**

**ج)** بر اساس پاسخ خود به قسمت (ب)، محتوای اولین بسته RREP را با بهره‌گیری از فرمت کلی بسته‌های RREP در شکل زیر تکمیل نمائید.

|  |  |  |  |
| --- | --- | --- | --- |
| مسیر ثبت شده[[10]](#footnote-10) | شناسه درخواست یکتا | گره مقصد درخواست مسیر | گره مبدا درخواست مسیر |
| S-B-D | 100 | D | S |

**د)** آیا RREP های دیگری هم توسط گره‌های دیگر به مبدا برگردانده می‌شوند. در صورت پاسخ مثبت، محتوای این RREPها را با بهره‌گیری از فرمت داده شده در قسمت (ج) به ازای هر یک رسم کنید. در صورت پاسخ منفی، استدلال خود را بیان نمائید.

**پاسخ:**

نه دیگه نباید پاسخ‌های دیگری به مبدا ارسال شود و بقیه گره‌ها از طریق overhearing باید بفهمن که پاسخی به مبدا ارسال شده بوده و کارش راه افتاده.

**ه)** در مورد الگوریتم معروف زمان برگشت نمایی[[11]](#footnote-11) تحقیق کنید. طریقه عملکرد این الگوریتم را با استفاده از شبه کد[[12]](#footnote-12) بنویسید و یک مثال معروف از کاربرد این الگوریتم در شبکه را بیان نمائید.

// Set initial variables

max\_attempts = 5

base\_delay = 1 // in seconds

attempt = 0

while attempt < max\_attempts:

attempt += 1

// Calculate delay time using the exponential backoff formula

delay\_time = base\_delay \* (2 \*\* (attempt - 1))

try:

// Make API call or perform some other action

response = make\_api\_call()

// If successful, break out of loop and return response

break

except Exception as e:

// Handle exception and log error message

log\_error\_message(e)

// Sleep for calculated delay time before attempting again

sleep(delay\_time)

// Return response or raise exception if max attempts have been reached

if attempt == max\_attempts:

raise Exception("Max attempts reached, cannot complete action")

else:

return response

**پاسخ:**

یک کاربرد بزرگی که این الگوریتم داره توی rate-limiterها است که مثلا درخواست‌ها رو میتونه با این روش محدود کنه و اگر هم زیادتر شد به صورت نمایی بازم زمان retry کردن رو عقب بندازه. کاربرد دیگر هم میتونه توی خطاهای شبکه باشه که باید با یه زمانی retry صورت بگیره و از این روش نمایی برای ریترای کردن بهره گرفته بشه.

**و)** بسته‌های داده موجود در بافر ارسال گره S را به تفکیک در زمان t = 2.9، t = 3.9 و t = 5.1 با ذکر شماره بسته و زمان ورود هر بسته در بافر ارسال روی شکل زیر مشخص کنید.

|  |  |  |
| --- | --- | --- |
| 3 | 2 | 1 |

Send buffer **S** (t=2.9)

|  |  |  |
| --- | --- | --- |
| 4 | 3 | 2 |

Send buffer **S** (t=3.9)

|  |  |  |
| --- | --- | --- |
|  |  |  |

Send buffer **S** (t=5.1)

**ز)** بصورت حداقل چند پیام RREQ از گره S ارسال می‌شود.

**پاسخ:**

فقط یدونه ارسال می‌شود چون وقتی مبدا می‌بینه به مقصدهای یکسانی برای بسته‌های مختلف احتیاج داره دیگه دوباره درخواست اضافی نمیفرسته. بعد از مدتی هم که دیگه ریسپانس یکی از درخواست‌ها میاد و توی کش ذخیره میشه و از همون استفاده میشه.

**نگهداری مسیر در DSR**

**سوال دوم -** یکی از چالش های پروتکل DSR وابستگی آن به کل مسیر است، بطوریکه شکستگی حتی یک لینک در کل مسیر می‌تواند بسیار آسیب‌رسان باشد.

**الف)** شکستگی یک لینک در چه شرایطی ممکن است اتفاق بیفتد؟

**پاسخ:**

اتفاقات مختلفی ممکنه بیفته و کلا هم شکستگی لینک توی شبکه‌های بی‌سیم از باسیم بیشتر رخ میده.

مثلا لینک دچار آسیب فیزیکی بشه. تداخل یا نویز به قدری زیاد بشه که دیگه لینک در دسترس نباشه. یه نود از نظر مکانی از محدوده خارج بشه. همچنین ممکنه به خاطر مشکل در باتری و تامین انرژی هم یک لینک دچار شکستگی بشه.

**ب)** برای اینکه شکستگی لینک اثر کمتری بر عملکرد پروتکل بگذارد، یک راه حل می تواند در نظر گرفتن پیش‌بینی زمان شکستگی در فرآیند کشف مسیر باشد. راه حل­های پیشنهادی شما چیست؟

**پاسخ:**

بخشی از راه حل به نظرم توی مانیتورینگ خوب از شبکه است یعنی اگر ما وضعیت شبکه و نودها مثل سیگنال دریافتی رو مانیتور کنیم می‌تونیم پیش‌بینی کنیم که چون این نود وضعیت خوبی نداره پس احتمالا خواهد شکست. البته این مانیتور کردن میتونه از وضعیت محیطی هم باشه که مثلا بدونیم نویز داره زیاد میشه.

اگر مسیرهای جایگزین رو هم ذخیره کنیم، وقتی که میفهمیم یه نودی شکسته می‌تونیم سریعا از مسیر جدید استفاده کنیم.

علاوه بر مکانیزم‌های روت‌مینتننس می‌تونیم خودمون هم به صورت دوره‌ای وضعیت نودها رو چک کنیم (یه جورایی همون مانیتورینگ میشه) که هزینه‌ی شکستن لینک‌ها رو کاهش بدیم.

**مکانیزم­های اضافه­شده در کشف مسیر**

**سوال سوم -** سربار[[13]](#footnote-13) ایجاد شده (تعداد بسته‌های درخواست مسیر) در مکانیزم Expanding Ring Search مورد استفاده در پروتکل DSR را در شرایط میانگین و همچنین بهترین و بدترین شرایط بررسی کنید.

**پاسخ:**

در بهترین حالت که همون همسایه‌ی اول پاسخ رو داره و جواب میشه برابر **۱** میشه.

در بدترین حالت باید به صورت مرحله به مرحله محدودیت رو زیاد کنیم تا به قطر شبکه برسیم.

اگر یکی یکی محدودیت رو زیاد کنیم، جمع یک دنباله‌ی حسابی میشه (۱ + ۲ + ... + قطر) که میشه برابر: D (D + 1) / 2

اما اگر محدودیت رو به صورت نمایی زیاد کنیم و هی ۲ برابرش کنیم، جمع یک دنباله‌ی هندسی میشه (۱ + ۲ + ۴ + ... + قطر) که میشه برابر: D2 – 1

پس در نهایت در حالت میانگین هم برابر میانگین حالت‌های بالا خواهد بود. یعنی اگر دنباله‌ی هندسی را در نظر بگیریم، میانگین برابر D2/2 می‌شود.

1. Cache [↑](#footnote-ref-1)
2. Route request hop limit [↑](#footnote-ref-2)
3. Per-hop propagation delay [↑](#footnote-ref-3)
4. Constant Bit Rate (CBR) [↑](#footnote-ref-4)
5. Send buffer [↑](#footnote-ref-5)
6. Unique request ID [↑](#footnote-ref-6)
7. Route Request [↑](#footnote-ref-7)
8. Route Reply [↑](#footnote-ref-8)
9. Preventing RREP storms [↑](#footnote-ref-9)
10. Route record [↑](#footnote-ref-10)
11. Exponential back off [↑](#footnote-ref-11)
12. Pseudo code [↑](#footnote-ref-12)
13. Overhead [↑](#footnote-ref-13)