BSM307 - Güz 2025 - Proje Önerisi
QoS Odaklı Çok Amaçlı Rotalama için Meta-Sezgisel ve Pekiştirmeli Öğrenme Yaklaşımları

Doç. Dr. Evrim GÜLER

İçindekiler

1 Projenin Amacı ve Kapsamı

Bu projenin temel amacı, 250 düğümlük karmaşık bir ağ topolojisi üzerinde, bir kaynaktan (S) bir hedefe (D) giden "en iyi" yolu bulmaktır. "En iyi" yol, tek bir kritere göre değil, üç farklı ve genellikle birbiriyle çelişen metrik üzerinden tanımlanacaktır:

1. **En Az Gecikme (Minimum Delay):** Verinin S'den D'ye en hızlı şekilde ulaşması.
2. **En Yüksek Güvenilirlik (Maximum Reliability):** Yolun (düğümler ve bağlantılar dahil) arızalanma olasılığının en düşük olması.
3. **En Az Ağ Kaynağı Kullanımı (Minimum Resource Usage):** Yüksek bant genişliğine sahip bağlantıların tercih edilmesi.

Bu problem, NP-Hard (NP-Zor) sınıfında bir problem olduğundan, geleneksel (deterministik) algoritmalarla verimli bir şekilde çözülemez. Bu nedenle, projenin kapsamı, bu optimizasyon problemini çözmek için **en az iki** farklı meta-sezgisel veya pekiştirmeli öğrenme algoritması geliştirip karşılaştırmaktır.

Datacenter Ağları Açısından Önemi Veri merkezleri (datacenter) ağlarında düşük gecikme, yüksek bant genişliği ve arıza toleransı kritik önemdedir. Trafik paterni yoğun Doğu–Batı (east–west) akışlarla karakterize olduğundan, darboğazların önlenmesi, akıllı yol seçimi ve yüksek güvenilirlik seviyeleri (ör. çoklu yollar, hızlı yeniden yönlendirme) hizmet sürekliliğinin temelini oluşturur. Bu proje, söz konusu gereksinimlere yönelik metrikleri (gecikme minimizasyonu, güvenilirlik maksimizasyonu ve kaynak kullanımını azaltma) doğrudan ele alır.

Bulut Ağları (Cloud) Açısından Önemi Bulut ağlarında esneklik, ölçeklenebilirlik ve çok kiracılı (multi-tenant) ortamda kaynakların verimli paylaşımı ön plandadır. Bağlantıların kapasite kısıtları, VM'ler/Kapsayıcılar arası trafiğin dinamikliği ve farklı hizmet seviyeleri (SLA) altında gecikme ve güvenilirlik hedeflerinin sağlanması karmaşık bir optimizasyon problemidir. Bu proje, bulut ve veri merkezi senaryolarında yaygın karşılaşılan bu zorlukları modelleyip, algoritmaların gerçekçi kısıtlar altında nasıl performans gösterdiğini gözlemleme fırsatı sunar.

2 Ağ Modeli ve Metriklerin Tanımlanması

İlk adım olarak, ağ topolojisinin rastgele (random) olarak oluşturulması gerekmektedir.

2.1 Ağ Topolojisi

- **Düğüm (Node) Sayısı (N):** 250
- **Bağlantı Olasılığı (P):** 0.4 (Erdős–Rényi G(n, p) modeli)
- **Gereksinim:** Oluşturulan grafiğin *bağlı* (connected) olduğundan emin olunmalı veya en azından S-D çiftleri arasında bir yolun var olduğu garanti edilmelidir.

2.2 Düğüm (Node) Özellikleri

Her bir `i` düğümü için:

- **İşlem Süresi ($ProcessingDelay_i$):** [örn: 0.5 ms - 2.0 ms] arasında rastgele bir değer.
- **Düğüm Güvenilirliği ($NodeReliability_i$):** [0.95, 0.999] arasında rastgele bir değer.

2.3 Bağlantı (Link) Özellikleri

Her bir `(i, j)` bağlantısı (kenar) için:

- **Bant Genişliği ($Bandwidth_{ij}$):** [100 Mbps, 1000 Mbps] arasında rastgele bir değer.
- **Gecikme ($LinkDelay_{ij}$):** [3 ms, 15 ms] arasında rastgele bir değer.
- **Bağlantı Güvenilirliği ($LinkReliability_{ij}$):** [0.95, 0.999] arasında rastgele bir değer.

3 Optimizasyon Metriklerinin Hesaplanması

Bir `S` düğümünden `D` düğümüne giden herhangi bir `P` yolu (Path) için (P= $\{n_1, n_2, ..., n_k\}$ düğümlerinden oluşur), üç metrik şu şekilde hesaplanacaktır:

3.1 Toplam Gecikme (Minimizasyon)

Gecikme *toplamsal* (additive) bir metriktir.

$$TotalDelay(P) = \sum_{(i,j)\in P} LinkDelay_{ij} + \sum_{k\in P, k\neq S,D} ProcessingDelay_k$$

(Yoldaki tüm bağlantı gecikmelerinin toplamı + *ara düğümlerdeki* (kaynak `S` ve hedef `D` hariç) tüm işlem gecikmelerinin toplamı)

3.2 Toplam Güvenilirlik (Maksimizasyon)

Güvenilirlik *çarpımsal* (multiplicative) bir metriktir.

$$TotalReliability(P) = \prod_{(i,j)\in P} LinkReliability_{ij} \times \prod_{k\in P} NodeReliability_k$$

Önemli Not: Çoğu optimizasyon algoritması toplama ile daha iyi çalışır. Bu metriği bir minimizasyon problemine dönüştürmek için **Güvenilirlik Maliyeti** (Reliability Cost) kullanılması tavsiye edilir:

$$ReliabilityCost(P) = \sum_{(i,j)\in P} [-\log(LinkReliability_{ij})] + \sum_{k\in P} [-\log(NodeReliability_k)]$$

(Bu değeri **minimize etmek**, toplam güvenilirliği **maksimize etmek** ile matematiksel olarak eşdeğerdir.)

3.3 Ağ Kaynak Kullanımı (Minimizasyon)

"En az kaynak kullanımı" metriğini, yüksek bant genişliğine sahip yolları tercih eden bir maliyet fonksiyonu olarak tanımlayabiliriz. Düşük bant genişliği "maliyetli" olmalıdır.

$$ResourceCost(P) = \sum_{(i,j)\in P} \left(\frac{1 \text{ Gbps}}{Bandwidth_{ij}} \right)$$

(Her bağlantının maliyeti, maksimum olası bant genişliğine (1 Gbps) göre ters orantılıdır.)

4 Proje Görevleri ve Gereksinimleri

Her grup (en fazla 6 kişi) aşağıdaki adımları tamamlamalıdır:

1. **Ağ Oluşturucu:** Yukarıda tanımlanan özelliklere göre rastgele bir ağ topolojisi oluşturan bir modül yazılmalıdır.
2. **Algoritma Seçimi:** Önerilen listelerden **en az iki (2)** farklı algoritmik yaklaşım seçilmeli ve uygulanmalıdır.
3. **Optimizasyon Çalışması:** Öğrenciler, **en az iki** metrik için optimizasyon sonuçları bulmalıdır ya da sadece aşağıda belirtilen çok amaçlı çözümü yapmalıdır.
4. **Çok Amaçlı Çözüm:** Üç metrik arasındaki dengeyi kurmak için **Ağırlıklı Toplam Yöntemi (Weighted Sum Method)** kullanılması tavsiye edilir:
   - Bir `P` yolu için "Uygunluk (Fitness)" veya "Maliyet (Cost)" fonksiyonu:

$$\begin{aligned} TotalCost(P) = & W_{delay} \cdot TotalDelay(P) + \\ & W_{reliability} \cdot ReliabilityCost(P) + \\ & W_{resource} \cdot ResourceCost(P) \end{aligned}$$

   - Burada $W_{delay} + W_{reliability} + W_{resource} = 1$ olacak şekilde ağırlıklar (W) belirlenir.
   - Öğrenciler, bu ağırlıkları değiştirerek farklı optimizasyon sonuçları bulmalıdır.
5. **Görsel Uygulama (Her algoritma için):**
   - Oluşturulan N=250 düğümlü grafiği görselleştiren bir arayüz.
   - Kullanıcının S ve D düğümlerini seçebilmesi.
   - Kullanıcının optimizasyon ağırlıklarını (W_{delay}, $W_{reliability}$, $W_{resource}$) ayarlayabilmesi.
   - "Hesapla" butonuna basıldığında, seçilen algoritmanın bulduğu en iyi yolu grafik üzerinde renkli olarak göstermesi.
   - Bulunan yolun metriklerinin (Toplam Gecikme, Toplam Güvenilirlik, Kaynak Maliyeti) ekranda gösterilmesi.

5 Önerilen Algoritmik Yaklaşımlar

5.1 Meta-Sezgisel (Meta-Heuristic) Algoritmalar

- **Genetik Algoritmalar (Genetic Algorithms - GA):**
  - **Çözüm (Kromozom):** S'den D'ye giden bir yol (örn: `[S, n5, n82, D]`).
  - **Uygunluk Fonksiyonu:** Yukarıda tanımlanan `TotalCost(P)` fonksiyonu.
  - **Operatörler:** Çaprazlama (Crossover) ve Mutasyon (Mutation).
- **Karınca Kolonisi Optimizasyonu (Ant Colony Optimization - ACO):**
  - "Karıncalar" S'den D'ye rastgele yollarla giderler ve düşük maliyetli yollara "feromon" bırakırlar.
- **Parçacık Sürüsü Optimizasyonu (Particle Swarm Optimization - PSO):**
  - Her "parçacık" potansiyel bir yolu temsil eder ve sürünün en iyi çözümüne doğru "hareket eder".
- **Benzetimli Tavlama (Simulated Annealing - SA):**
  - Tek bir çözümle başlar ve bu yolu küçük "mutasyonlarla" geliştirir.
- **Yapay Arı Kolonisi (Artificial Bee Colony - ABC):**
  - Bal arılarının zengin nektar kaynaklarını (yiyecek kaynakları) akıllıca bulma ve paylaşma davranışını taklit eder.
- **Değişken Komşuluk Araması (Variable Neighborhood Search - VNS):**
  - "Bir komşulukta takılırsan, komşuluğun şeklini değiştir" fikrine dayanan çok güçlü ve zarif bir yöntemdir..

5.2 Pekiştirmeli Öğrenme (Reinforcement Learning - RL) Algoritmaları

- **Q-Learning (Q-Öğrenme):**
  - **Durum (State):** Ajanın şu anda bulunduğu düğüm `i`.
  - **Eylem (Action):** `i` düğümünden komşu bir `j` düğümüne gitmek.
  - **Q-Değeri (Q(s, a)):** `s` durumundayken `a` eylemini yapmanın beklenen gelecekteki toplam ödülü.
  - **Ödül (Reward):** Hedefe ulaşıldığında, bulunan yolun `TotalCost(P)` değerine göre (örn: $Reward = 1000/TotalCost(P)$) verilir.
- **Sarsa:**
  - Q-Learning'e benzer ancak politika-içi (on-policy) bir algoritmadır.

6 Deney Düzeneği ve Kıyaslama

- En az 20 farklı (S, D, B) örneği; başarısız/uygunsuz örnekler gerekçeleriyle raporlanmalıdır.
  - S: Kaynak düğüm
  - D: Hedef düğüm
  - B: Talep edilen bant genişliği
- En az 2 farklı algoritma kıyaslama.
- En az 5 tekrar yapılıp: ortalama, standart sapma ve en iyi–en kötü sonuçların raporu.
- Çalışma süresi (zorunlu), ölçeklenebilirlik analizi (opsiyonel).

7 Teslim Edilecekler

1. **Proje Raporu (PDF):** **Son Teslim: [7 Ocak 2026 Saat 23:59]**
   - Problemin tanımı ve kullanılan ağ modeli.
   - Seçilen algoritmaların (en az 2) teorik açıklaması.
   - Uygulamanın mimarisi ve kullanılan teknolojiler.
   - **Karşılaştırmalı Sonuçlar:** Farklı ağırlıklar için elde edilen sonuçların analizi ve algoritmaların performans karşılaştırması.
2. **Kaynak Kodları:** İyi yorumlanmış (commented) kaynak kodları: Git deposu, README, seed bilgisi ve çalıştırma adımları.
   Son Teslim: [31 Aralık 2025 Saat 23:59]
3. **Çalıştırılabilir Uygulama ve Demo:** Projenin çalıştığını gösteren kısa bir video kaydı, sınıfta canlı sunum ve soru–cevap.
   Son Teslim: [31 Aralık 2025 Saat 23:59]

8 Değerlendirme Rubriği

- Doğruluk & Kısıtlara Uyum – %20
- Algoritmik Çeşitlilik – %30
- Performans – %10
- Görselleştirme & Uygulama – %15
- Deney Tasarımı – %15
- Raporlama & Sunum – %10
- Ek puan: Çok-amaçlı Pareto, GNN-RL, >1000 düğüm, ILP karşılaştırması

Pareto Optimalite Nedir? Pareto Optimal (veya Pareto Verimli) bir çözüm, diğer herhangi bir çözüme göre baskın olmayan (non-dominated) bir çözümdür. Daha basit bir ifadeyle: Bir çözüm Pareto Optimal ise, o çözümün hedeflerinden (örn. gecikme) herhangi birini iyileştirmek için, diğer hedeflerden (örn. güvenilirlik) en az birini feda etmek (kötüleştirmek) zorunda kalırsınız.

Projemizden Örnek: Diyelim ki iki yol buldunuz:

- Yol A: Gecikme$= 10ms$, Güvenilirlik$= 0.99$, Maliyet$= 50$
- Yol B: Gecikme$= 12ms$, Güvenilirlik$= 0.999$, Maliyet$= 45$

Hangi yol daha iyi seçimdir? Cevap: "Duruma göre değişir." Yol B, Yol A'dan daha güvenilir ve daha az maliyetli, ancak daha yavaş.

Bu iki çözüm birbiriyle kıyaslanamaz (biri diğerine her açıdan üstün değil). İkisi de potansiyel olarak "Pareto Optimal" olabilir.

Şimdi üçüncü bir yol ekleyelim:

- Yol C: Gecikme$= 15$ms, Güvenilirlik$= 0.98$, Maliyet$= 55$

Yol C'yi analiz edelim: Yol A ile kıyaslandığında, Yol C her açıdan daha kötüdür (daha yavaş, daha az güvenilir, daha maliyetli). Bu durumda diyebiliriz ki: "Yol A, Yol C'ye baskındır (dominates)". Bu nedenle, Yol C asla optimal bir çözüm olamaz ve Pareto Sınırı'na dahil edilmez.