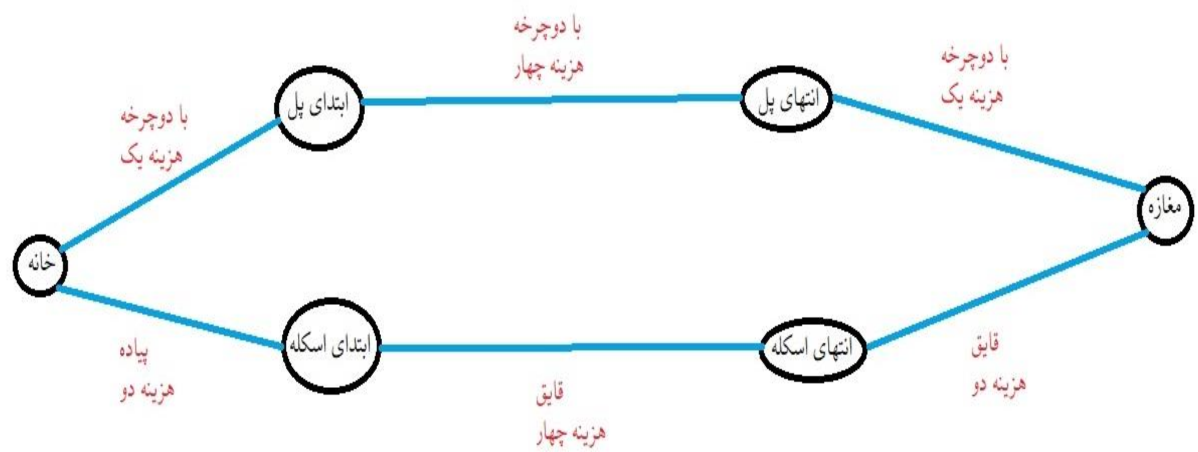
با توجه به گفته‌های صورت سوال، می‌توانیم فاصله هر خانه تا اسکله و یا پل و همچنین فاصله اسکله و یا پل را بصورت یک گراف بدون جهت(در حالت کلی چرا که در صورت سوال به یکطرفه بودن هر خانه تا پل و یا اسکله و هر پل یا اسکله تا مغازه اشاره‌ای نشده است) مدل کنیم. در این گراف از هر راس خانه تا پل یک یال با وزن یک قرار میدهیم چرا که طبق گفته صورت سوال، فاصله خانه تا پل با دوچرخه برابر یک لگو است. و همچنین یک یال از هر خانه تا اسکله قرار میدهیم(ابتدا و انتهای پل و یا اسکله را هر کدام با یک راس در گراف‌مان مدل می‌کنیم.) و فاصله آن را برابر دو قرار می‌دهیم. همچنین فاصله ابتدا تا انتهای پل/اسکله را با یک یال با وزن چهار مقداردهی می‌کنیم. تحلیل مشابهی برای تکمیل گراف در سمت راست داستان ما نیز برقرار است.

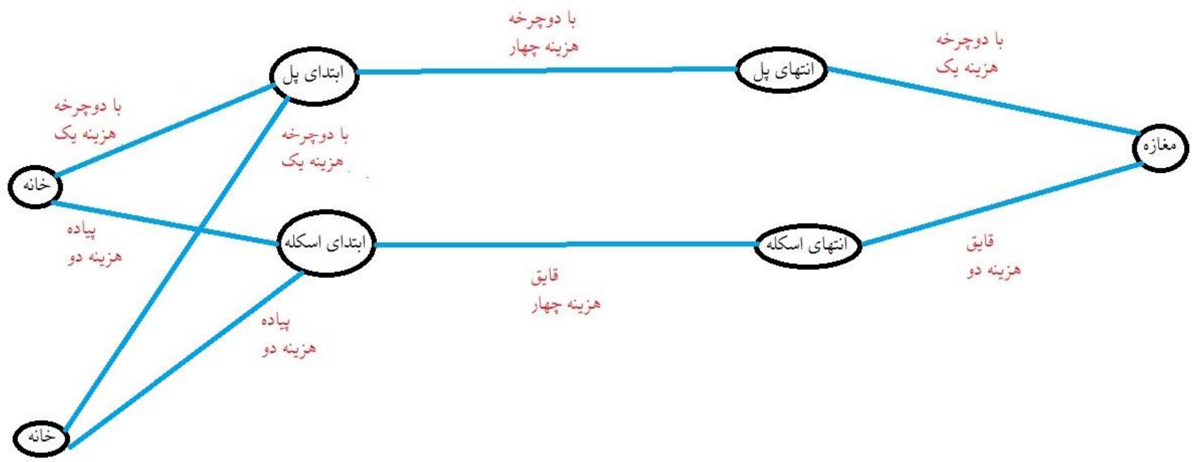
شکل زیر گراف مدل شده را در ساده‌ترین حالت که یک خانه و یک مغازه داریم نشان می‌دهد.



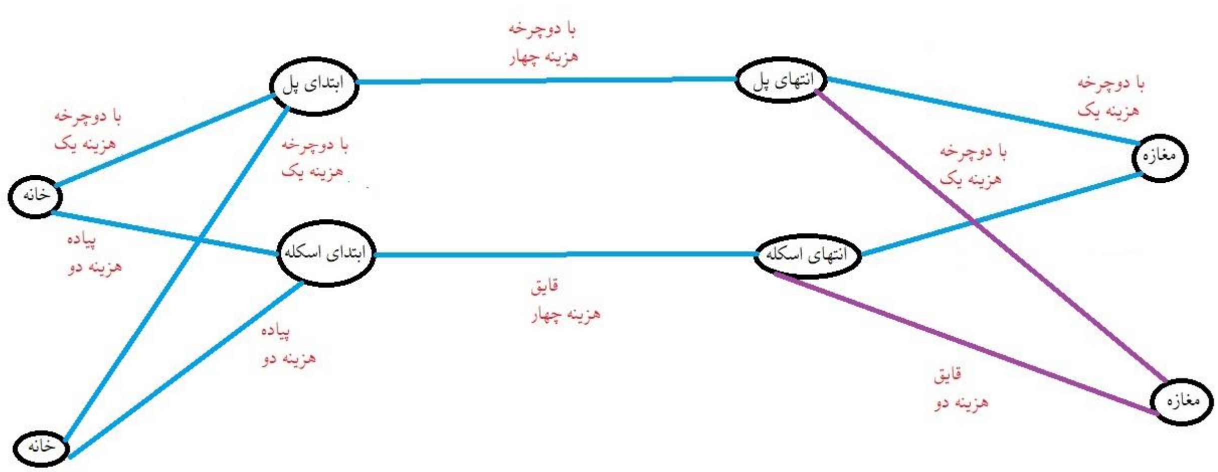
حال میتوان این گراف را گسترش داد و هر تعداد خانه و یا مغازه و پل و اسکله را به آن افزود.

نکته مهم اینست که توضیحات صورت سوال راجع به فاصله هر خانه تا هر پل /اسکله و فاصله هر پل/اسکله تا مغازه اندکی مبهم است. ما در اینجا فرض می‌کنیم که فاصله تا پل با دوچرخه برابر یک و تا اسکله با پای پیاده برابر دو است و طبق این فرض پیش می‌رویم هر چند که فرضیات دیگر خللی به توضیحات ما وارد نمی‌کند.

حال با گسترش گراف به ازای هر پل/اسکله نیاز به دوراس و یک یال بین آن‌ها داریم. در اینجا می‌توان کمی هوشمندانه عمل کرد تا نیازی به دو راس به ازای هر پل/اسکله نباشد. بدین صورت که یال هر خانه را به یک پل/اسکله متصل کنیم. شکل زیر شمایی از دو خانه و یک پل را نشان می‌دهد.



چنین کار مشابهی را میتوان برای سمت چپ رودخانه نیز بکار برد که در شکل زیر آمده است:



حال برای یافتن کوتاه‌ترین فاصله هر خانه تا تمام مغازه‌ها می‌توان الگوریتم‌های یافتن کوتاه‌ترین مسیر گراف را بکار برد. باتوجه به اینکه تمام کوتاه‌ترین مسیرها را می‌خواهیم، بایستی الگوریتم‌های

all-pair shortest path را بکار ببریم. پیشنهاد اول الگوریتم شبه ضرب ماتریسی است که مرتبه $O(n^4)$ دارد که می‌توان آن را به $0(n^3\ log\ n)$ کاهش داد. الگوریتم بهینه‌تر برای انجام این کار الگوریتم فلوید وارشال است که مرتبه‌اش $0(n^3)$ است.در حالت کلی و برای گراف‌های شلوغ(منظور از شلوغ این است که تعداد یال‌ها برابر $O(n^2)$ باشد.) راه‌حلی بهتر از این مرتبه مکعبی وجود ندارد اما برای گراف های خلوت(منظور از خلوت این است که تعداد یال‌ها برابر O(n) باشد.) و یا گراف‌هایی که حداکثر مقدار یال‌هایش یک مقدار ثابت است، الگوریتم‌هایی بهتر از مرتبه مکعبی وجود دارد. یکی از این الگوریتم‌ها الگوریتم جانسون است که از توضیح جزئیات آن صرف نظر می‌کنیم و فقط به این نکته اشاره می‌کنیم که این الگوریتم از دو زیر روال الگوریتم دیکسترا و بلمن فورد و تکنیک وزن دهی مجدد(در صورتی که وزن یال‌ها منفی باشد) استفاده می‌کند و برای گراف‌های خلوت مرتبه‌ی زمانی الگوریتم جانسون برابر $0(n^2\ log\ n)$ است که خب بهتر از الگوریتم فلوید وارشال است.

حال باتوجه به اینکه گراف ما یک گراف خلوت است می‌توانیم از الگوریتم جانسون استفاده کنیم.

نکته خیلی مهم این است که ما باز هم میتوانیم مرتبه را کاهش دهیم. راه‌حل استفاده از BFS یا همان جستجوی سطحی است. جستجوی سطحی برای گراف‌هایی که وزن یکسان دارند، درخت کوتاه‌ترین مسیر را می‌دهد. در اینجا مشکلی که وجود دارد این است که وزن یال‌ها متفاوت است اما این مشکل را می‌توان با افزودن تعدادی راس به عنوان راس dummy برطرف کرد و وزن تمام یال‌ها را به یک رساند و سپس بر روی گراف حاصل جستجوی سطحی را اعمال کرد. مرتبه‌ی زمانی جستجوی سطحی O(n+e) است که چون در گراف ما تعداد یال‌ها هم‌مرتبه‌ی تعداد راس‌هاست پس در نتیجه مرتبه‌ی زمانی جستجوی سطحی در گراف ما برابر O(n) خواهد شد که این مرتبه‌ی زمانی خودش یک کران پایین است و بهتر از O(n) نمی توان این مسئله را حل نمود.