Algorithmic Approach to 15 Minute City

Marco Lam

April 29, 2024

Abstract

The 15-Minute City is an urban planning concept introduced in the last decade that promotes accessibility. It posits that residents should be able to meet basic needs—such as groceries, education, healthcare, and leisure—within a 15-minute travel time from their homes. The concept aims to deliver environmental, social, and economic benefits by reducing reliance on automobiles, encouraging active transportation, and enhancing residents' quality of life through easy access to essential services and amenities. The concept has gained traction during the COVID-19 pandemic, which hightlighted the significance of local services and amenities in urban settings.

The 15-Minute City concept has been explored across various research fields, including urban planning, transportation, and environmental science. However, a general methodology to identify areas of a city that qualify as a 15-Minute City is lacking. Most existing studies are data-driven, focusing on specific cities with solutions that are often neither algorithmic nor generalized.

This thesis aims to develop a general, adaptive, and efficient algorithm to identify city areas that can be classified as a 15-Minute City. It examines several existing algorithms for graph data structures, such as Breadth-First Search, Dijkstra's algorithm, Johnson's algorithm, and their variations. The proposed "15-Minute City algorithm" synthesizes ideas and techniques from these algorithms to offer a comprehensive and efficient solution for determining 15-Minute City areas.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

The 15-Minute City was first proposed by Moreno in 2016 as a solution to build safer, more resilient, sustainable and inclusive cities and to harmonise the notion of Smart Cities [MAC+21]. Moreno et al. discussed the relationship between the concept of the "15-Minute City" to Urban Planning Pandemic Response, its Emerging Variations (i.e. 20 minute city [CDSKL19]), walkable neighbour [WDL+19] smart cities.

Moreno et al. proposed the formal "15-Minute City Concept", arguing that residents should be able to enjoy a higher quality of life where they will be able to effectively fulfil six essential urban social functions to sustain a decent urban life. These include (a) living, (b) working, (c) commerce, (d) healthcare, (e) education and (f) entertainment. The authors then proposed the "modified 15-Minute City" framework, depicting the four identified dimensions that could be incorporated with the already existing one proposed. These are (a) Density, (b) Proximity, (c) Diversity and (d) Digitisation.

Since COVID-19 pandemic, this topic has gained a tremendous amount of attention and has growth exponentially in popularity [LC23, ABCM22] among various research fields in literature, with urban design being the most studied [LC23]. The concept of the 15-Minute City has been studied and shown that it brings benefits to the society (environmental, social, and economical etc.) [ABCM22].

However, many of the studies in 15-Minute City (or t minute city in general) in the literature have used a data-driven approach. Although the 15 minutes threshold is attracted the most attention, the notion of the t-Minute City has also been considered [MAC+21]. Some examples are the 20-Minute City in Tempe, Arizona [CDSKL19], the 30-Minute City in Sydney, Australia [SWL20] and Olivari et al. studied 5 to 60 Minute City in Ferrara, Italy [OCNG23]. The lack of algorithmic approaches available regarding t minute city means that many solutions available cannot be generalised and to be reapplied to another location of interest. Moreover, Lima et al. has noted that computational approaches to 15-Minute City design presents significant challenges such as "data availability and quality", "computational cost" and "adaptability" [LC23]. Notably, Weng et al. [WDL+19] and Olivari et al.[OCNG23] relied on census data on population density for their respective studies which may not be available for some countries and regions.

Chapter 2

Previous Work

In this section, we will discuss a number of previous works of 15-Minute City. As this topic have been studied in various research fields in literature, we will group the works according to their methodologies and approaches, which include graph representation, grid tessellation, flow data, Walk Score, and other works. Table 2.1 provides a summary of these studies mentioned in this section excluding other works, specifically the methodologies discuessed and used in determining and finding the so-called "15-Minute City". Other works will cover studies about the general concept of the "15-Minute City". The following sections are the detailed descriptions of each study and methodology used.

Table 2.1: Summary of relevant previous works

Approach	Work	Description
Graph Representation	[BDMM23], [CCRZ22], [RSRBH23]	Maps represented by graphs as a mathematical structure
Grid Tessellation	[GGZC22], [OCNG23], [PAMC24]	Maps divided according to various shapes and 15-MC calculations are applied to each area independently
Flow Data	[ZZK+23], [SWL20]	Use foot travel data to incorporate human mobility patterns
Walk Score	[Wal], [WDL+19]	Proprietary methodology based on sets of specific factors

2.1 Graph Representation

2.1.1 Graph Representation of the 15-Minute City: A Comparison between Rome, London, and Paris

Barbieri et al. defined the general t-minute city (hereafter t-MC) on an urban graph with respect to a given set of services [BDMM23]. The urban graph is represented by a planer graph $G(V, E)$ (when a connected graph can be drawn without any edges crossing).

The nodes of G are the intersections of the roads, and the lengths of the edges are proportional to the travel time with a coefficient, which is the speed of the pedestrians. Services $f \in V$ are then placed by adding an extra node to the graph, or use the nearest junction (as per Euclidean distance).

Given a list of N_i services of type i, $C_k^i = (C_1^i, C_2^i, ..., C_{N_i}^i)$ are the nodes that reach f^i in less than 15 min. The set of vertices that form a 15-MC with respect to services of type i is given by the union of all these vertices

$$C^i = \bigcup_{k=1}^{N_i} C_k^i \subseteq V$$

If services f_i, f_j of the same type are far enough, it is possible that $C_i \cap C_j = \emptyset$. The authors noted that these "gaps" can be recognised as "the places where it is necessary to intervene to reconnect". Then for K types of services, the set of the 15-minute vertices is

$$C = \bigcap_{i=1}^{K} C^i = \bigcap_{i=1}^{K} \left(\bigcup_{j=1}^{N_i} C_k^j \right) \subseteq V$$

Finally, the authors define G_C as the graph induced on G by the vertices C, the 15-MC graph is the subgraph G_C of the urban graph G. To formalise, the set C depends on the travel time t, the service matrix $\mathbf{s}$, and the travel speed v, the graph of the 15-MC can be defined as

$$G_C = G_C(C, E_C; t, \mathbf{s}, v) \subseteq G$$

A metric/ratio $\gamma(r, x_0; t, \mathbf{s}, v)$ was also defined as a function of the radius r with respect to a given origin x_0, which can be used to characterise the 15-MC and compare different cities or areas of the same city, where

$$\gamma(r, x_0; t, \mathbf{s}, v) \equiv |C(r, x_0; t, \mathbf{s}, v)|/|E|$$

A possible generalisation of the index was then suggested which takes into account the properties or weights $w(e), e \in E$ of each edge, such as the length of the path, the population density or the slope of the streets

$$\gamma = \frac{\sum_{c \in C} w(c)}{\sum_{e \in E} w(e)}$$

If $w(\cdot)$ is the population density, the edges of the parks and archaeological area have $w = 0$, and the index γ is not biased.

For the case study, the authors used a "shortest path search algorithm" for graph search [Dij59].

***Remark**: This paper transform map data into a planar graph, the algorithm starts from services rather than houses which reduces computational complexity. The authors referenced an article about Dijkstra algorithm for graph search but their implementation was not mentioned.*

2.1.2 Exploring the 15-minute neighbourhoods

According to the authors (Caselli et al.), "in the proposed study, the 15-Minute City theme is addressed with an analytical model designed and developed using GIS to assess exist-

ing conditions of accessibility to neighbourhood services for all the resident social groups." [CCRZ22]

The GIS-model is implemented by improving and integrating a Territorial Information system (managed with ArcGIS software). Extracting the pedestrian paths feature class to generate a link-node graph with all walking routes available. The model also considers that users "might choose to walk along road margins or cross in the proximity of road intersections."

The paper studied the area covered that can be travelled to "neighbour cores" within 15 minutes by the following calculation:

$$\text{Length}(km)/(3km/h \times 60min) + \text{DF}(min)$$

where DF is the delay factor at crossings (20 and 30 seconds for non-signalised and 40 and 60 seconds for signalised).

The authors then compare this area with its population distribution to study the proportion of population covered by the 15-MC.

Remark: *The authors did not define the neighbour cores, only by saying "urban nodes well served by necessities shops and services, such as supermarkets, grocery stores, bars, drugstores, and banks." However, using such nodes to calculate for 15-Minute City contributes to faster running time. The actual 15-Minute City search approach was not mentioned.*

2.1.3 The inclusive 15-Minute City: Walkability analysis with sidewalk networks

The paper proposed a framework for assessing multi-factor walkability on a sidewalk network model [RSRBH23]. The sidewalk network model is defined as a graph where the nodes are intersections or crosswalks, the edges have 3 types: sidewalks, crosswalks, and pedestrian-only paths and 4 attributes: length, width, slope, and pedestrian hazard. The pedestrian-only paths includes pedestrianised streets, living streets, and paths through parks and plazas. Pedestrian hazard is a metric to describe how dangerous each sidewalk segment by using a fine-grained map of estimated pedestrian safety in Barcelona [BRSR+21] and by exploiting Deep Learning tools. The resulting network is denser than the road network (approximately 4 to 1 in both nodes and edges). Each node of the graph has also been assigned a population according to census data.

This network is then simplified by a percolation analysis according to the sidewalks' properties (i.e., width, slope, or hazard). The authors then noted that the analysis on the 1260m that can be travelled in 15 min at a walking speed of 1.4 m/s, in accordance with literature [BW17]. Using government data, the authors selected a list of services.

To find such a set of links, we extended the classic Dijkstra algorithm to

1. explore all nodes within the threshold time from a single source, and
2. to record all edges that can be traversed within the threshold, not only the ones that form part of a shortest path.

A formal description can be found in Section S3 of the Supplementary Materials, while the implementation was done in Python, using the igraph library [igr].

The authors then studied the impact of population size of the largest and second largest connected component of Barcelona's sidewalk networks by changing the parameters of percolation analysis.

Remark: *This model takes hazard as a factor. The percolation analysis is a main focus of the study. The Supplementary notes also included the modification of Dijkstra algorithm.*

2.2 Grid Tessellation

2.2.1 Urban accessibility in a 15-Minute City a measure in the city of Naples, Italy

The authors (Gaglione et al.) in this paper proposed a 4 steps methodology through a GIS environment to define the areas accessible in 15 minutes [GGZC22].

1. With a systemic approach, 17 variables have been identified by
   - The characteristics of the population.
   - The characteristics of urban fabrics, in particular their shape.
   - The physical characteristics relating to safety, amenities and pleasantness of the pedestrian. network.
2. The relationships among different groups of characteristics were identified through a (Pearson) correlation analysis to remove some variables.
3. Relating the demand (users) to the supply (local urban services). The authors used a proximity analysis by calculating the Euclidean distances from the centroids of the census sections to the related closer local urban services, then study how users can move along the pedestrian network by labelling 13 characteristics on each pedestrian path. "On the basis of the travel times defined on each link of the pedestrian network and the distribution and location of all the local services examined, urban areas accessible in 15 minutes have been defined."
4. The population density is then compared with the 15-minute accessible areas. Individual age groups are also studied in this context.

The authors then explored the effect of choosing different grid size, shapes etc. in terms of the results of the minute city.

Remark: *The authors did not specify walking speeds or any algorithms used in calculating travel times.*

2.2.2 Are Italian cities already 15-minute?

The authors (Olivari et al.) of this paper proposed a data-driven approach solution by defining the NExt proXimity Index (NEXI), which "exploits the data to answer the question: "Which parts of your city or town already follow the 15-minute model?" [OCNG23]

With a selected list of service categories, and the nodes of the road network (the intersection points of the network geometries) and the points of interest (the geographical location of the various services). For each node the algorithm computes the time needed to reach – at an average walking speed - the closest point of interest of any given category, being constrained

to move only on roads accessible to pedestrians. More in details, the time needed to reach the points of interest is computed as $t = l/s$, where:

- l is the length of the shortest route to the PoI (Point of Interest), on a road network made only of walkable roads,
- s is the approximate walking speed of an average person, that is 5 km/h.

If all categories can be reached within 15 minutes, the node is then considered to be a 15-minute node. Using a 5 km/h speed, the maximum reachable distance in 15 minutes is 1250m. "if the average time to reach all the categories from the nodes in that area is lower or equal to 15 minutes".

The algorithm computes the level of proximity of a given area as the mean of the levels of proximity of the nodes inside that area. Therefore, an area is 15-minute if the average time to reach all the categories from the nodes in that area is lower or equal to 15 minutes. The authors used hexagons with a diameter of 250 meters as the smallest resolution unit.

3 indices are proposed by the authors:

1. The NEXI-Minutes assigns to each category for each area a value of time which is the average time to reach each category.
2. The NEXI-Global takes inspiration from the Walk Score methodology, measuring the global proximity to all service categories on a scale that goes from $0 - 100$, where 0 means that none of the categories is at least within a 30-minute walk, while 100 means that all categories are within a 15-minute walk and all values in between describe an intermediate situation.
3. A discomfort index which takes population into account, where

$$\text{Discomfort} = (100 - \text{Global}) \times \text{Population}$$

Remark: *This paper takes the population distribution into account, only considering walking (but can be changed easily by varying the speed parameter). It does not use graph representation and there are no mentions of graph search algorithm. Though the authors mentioned that the algorithm could be sped up by using "parallelisable algorithm".*

2.2.3 Travel-time in a grid: modelling movement dynamics in the "minute city"

The authors claimed to "provide initial insights and recommendations for developing more robust 15-Minute City models" and emphasised "the importance of technical modelling steps in determining the mapping outputs which support the assessments of 15-minute cities" [PAMC24]. In the paper, they experimented on evaluating grid-based methods and identified four noteworthy variables:

1. Grid tessellation choices
2. Software application pick
3. Speed selection for travel-time calculations
4. Classification rules' adoption for mapping urban functions against mapped amenities

"The absence of standardised modelling protocols imposes significant limitations on the application of MC models, hinders synchronic comparisons, and can indirectly foster the formulation of exclusive policies and inconsistent planning decisions. Hence, it is crucial to undertake concerted efforts towards standardisation, including by bridging the gap between the planning practice and software development communities, to effectively address the existing challenges in MC modelling and representation. This entails establishing shared modelling protocols, algorithms (across various software applications), and standardised data inputs, all of which can substantially enhance the consistency and reliability of grid-based MC assessments focused on travel-time estimates. Ultimately, advancing research, fostering collaboration, and promoting knowledge sharing, can elevate the quality of evidence generated through spatial analysis, leading to better-informed decisions in MC planning."

Remark: *The authors concluded the article with the paragraph above, which highlights the importance of having a standardised method in calculating 15-Minute City. The results of the thesis can be used to be a harmonised solution to this.*

2.3 Flow Data

2.3.1 Towards a 15-minute city: A network-based evaluation framework.

In the context of 15-Minute City, it is suggested that the allocation of facilities should account for how people access and use local service and amenities rather than merely considering population size [CLZ20].

The authors in this paper proposed a methodological framework for evaluating 15-Minute City based on network science approaches. The paper proposes a network-based approach to evaluating 15-Minute City [ZZK+23]. This approach differs from mostly used accessibility measurements by accounting for human mobility patterns.

The network-based evaluation framework contains 3 parts:

1. Optimal mobility network is estimated based on the spatial distribution of urban amenities and population using a maximum flow algorithm.
2. The actual origin-destination network is obtained using mobile phone signalling data.
3. The differences between actual origin-destination network and the optimal network are measured to provide insights on the extent to which human mobility patterns, as a reflection on the usage patterns of urban amenities, match or do not match the schemes of urban planning and construction.

The authors then applied the framework model to a case study in Nanjing, China.

2.3.2 Measuring polycentricity via network flows, spatial interaction and percolation

This paper studied polycentricity based on inflow and outflow trip data and considered 3 network-based centricity metrics [SWL20]. In particular, accessibility-based centricity index computes the total number of jobs available and the number of works available within a time threshold from a location. The time threshold was set to 30 in the study to align

with the polycentricity-inspired masterplan proposed by The Greater Sydney Commission (GSC) in 2018 which applies to the Sydney-GMR (Sydney Greater Metropolitan Region), noting that "access to jobs, goods and services is provided to the community in three largely self-contained regions."

2.4 Walk Score

2.4.1 Walk Score

Walk Score is a measure of how walkable an address is based on the distance and availability of nearby amenities, such as grocery stores, restaurants, schools, parks, etc. The higher the Walk Score, the more walkable the location is.

According to the Walk Score methodology [Wal], for each address, hundreds of walking routes to nearby amenities are analysed. Points are awarded based on the distance to amenities in each category. Amenities within a 5 minute walk (0.25 miles) are given maximum points. A decay function is used to give points to more distant amenities, with no points given after a 30 minute walk. Walk Score also measures pedestrian friendliness by analysing population density and road metrics such as block length and intersection density. Data sources include Google, Factual, Great Schools, Open Street Map, the U.S. Census, Localise, and places added by the Walk Score user community.

The Walk Score calculation can be summarised into 4 steps

1. Assigning raw weights for selected amenities
2. Calculating distances from each location (community, data from government) to the selected amenities
3. Computing the total scores based on the distances and modifying the scores according to decay factors (e.g. street intersections and block length)
4. Normalising scores to 0–100

Walk Score ranges from 0 to 100, with the following descriptions:

- 90–100: Walker's Paradise. Daily errands do not require a car.
- 70–89: Very Walkable. Most errands can be accomplished on foot.
- 50–69: Somewhat Walkable. Some errands can be accomplished on foot.
- 25–49: Car-Dependent. Most errands require a car.
- 0–24: Car-Dependent. Almost all errands require a car.

Remark: *The Walk Score algorithm is not open source. However, it considers walking distance between 5 to 30 minutes. Its calculation has been validated [CDM11]*

2.4.2 The 15-Minute Walkable Neighbourhoods: Measurement, Social Inequalities and Implications for Building Healthy Communities in Urban China

In this paper, the authors (Weng et al.) noted some of the limitations of the Walk Score calculation [WDL+19], such as

1. It targets at overall population and the walking demands of different pedestrian groups have not been included in the assessment.
2. The decay effect of amenity varies greatly among population groups and categories of amenities.
3. Actual traffic situation has not been considered when calculating distances based on Euclidean distance.

They proposed a modified method to measure 15-min walkable neighbourhoods based on the Walk Score metric, taking into account pedestrians' characteristics and amenity attributes (scale and category). 6 categories of amenities were studied, including education, medical care, municipal administration, finance and telecommunication, commercial services, and elderly care. They delivered the questionnaire to 132 respondents to conclude the parameters considered in the metric. They also used a decay factor to account for different age groups etc in terms of walking speed.

Remark: *This paper did not consider any algorithm and modification is heavily based on a questionnaire.*

2.5 Other works

2.5.1 A Grammar-Based Optimisation Approach for Designing Urban Fabrics and Locating Amenities for 15-Minute Cities

This paper uses a geometric grammar based approach to explore computation to support decision-making concerning the layout of urban fabrics and the location of amenities in a neighbourhood [LBD22]. The authors used an inductive method for qualitative content analysis. However, the authors noted that this solution "does not address irregular or non-orthogonal urban block patterns, and the influence of nearby amenities located outside the studied fabric was not considered."

2.5.2 The Quest for Proximity: A Systematic Review of Computational Approaches towards 15-Minute Cities

The authors developed a comprehensive overview of the use of computational tools to support the analysis and design of 15-Minute Cities using a systematic literature review [LC23]. They noted that the topic of 15-minute city has growth exponentially in popularity and especially in the field of urban design. They concluded that computational approaches to 15-minute city design presents significant challenges such as "Data availability and quality", "Computational cost" and "Adaptability".

2.5.3 The 15-minute city: Urban planning and design efforts toward creating sustainable neighbourhoods

The authors collected 103 documents, dealing with underlying principles, sustainability advantages, and critics of the 15-minute city concept [KGSS23]. They defined 7 dimensions constitute the 15-minute city: (1) proximity, (2) density, (3) diversity, (4) digitalisation, (5) human scale urban design, (6) flexibility, (7) connectivity. The authors summarised the sustainability contributions in social, economical and environmental aspects in the society by 15-Minute City and also the barrier of implementations.

2.5.4 The Theoretical, Practical, and Technological Foundations of the 15-Minute City Model: Proximity and Its Environmental, Social and Economic Benefits for Sustainability

15-Minute City has four main cornerstones (proximity, diversity, density, and digitisation). The authors in this paper explored the proximity dimensions of the 15-Minute City and how it could influence mixed land use to yield environmental, social, and economic benefits [ABCM22].

2.5.5 Urban Transition and the Return of Neighbourhood Planning. Questioning the Proximity Syndrome and the 15-Minute City

The authors of this paper developed an evidence-based approach to a deeper analysis of policy design and implementation of the 15 minute city [MB22]. They concluded that the implementation and effectiveness of the 15 minute city depend on the concrete and contextual conformation of each city. The authors ended the paper stating that city development "needs some design framework and structure capable of addressing transformations, and their space and time location and sequences" on top of the "15 minute device" and that "it needs a syntax for an urban planning course of action that is incremental and adaptive but not limited to the contingent, blurred, and agnostic appeal of a catchy label."

Chapter 3

Problem Statement

In the previous section 2, serveral studies have discussed the needs for standardisation in terms of the methodogy used in computing the 15-Minute City. In particular, Pezzica et al. argued that "the absence of standardised modelling protocols imposes significant limitations on the application of MC models, hinders synchronic comparisons, and can indirectly foster the formulation of exclusive policies and inconsistent planning decisions" [PAMC24]. Lima et al. noted that "introducing computational approaches to 15-minute city design presents significant challenges and potential bottlenecks. On the other hand, exploring these challenges as opportunities for inserting new research is also possible since the theme is rising", some of these challenges are "Data availability and quality", "Computational cost" and "Adaptability" [LC23]. Furthermore, Marchigiani et al. found that the approach to 15-Minute City needs to be changed and adapted to each location case by case [MB22].

With these in mind, the aim of this thesis is to provide a general, adaptable algorithm to identify the 15-Minute City, where a person can travel to all their needs within 15 minutes from their home. Formally, the algorithm has the following properties.

- Inputs
  1. A graph data structure $G(V, E)$ representing the entire area of which t-Minute City is computed. V is the set of vertices representing locations of the area, this could be road junction, amenity of interest, etc. E is the set of weighted, undirected edges such that $E \subseteq V \times V$ and the weights $w : E \to \mathbb{R}_+$ of the edges are proportional to the length of the roads, in minutes.
     - More specifically, given a list of services of p distinct types and their locations, each location can be inserted into the graph by adding a vertex to V along an existing edge in E, in such a way that the distance from the vertex to the nearest road junction is minimised.

     For every vertex $v \in V$, v has the label $v.l \in \{0, 1\}^p$, where $v.l[i]$ represents the availability of service type i of the location. For example, $v.l \in \{0\}^p$ can be intrepreted as a road junction.
  2. A time threshold t in minutes.
- Output

  A set of vertices $R \subseteq V$ of vertices which can reach at least one location of each service type within t minutes, such that

$$\forall v \in R : \forall i \in \{1, ...p\}, \exists w \in V, w.l[i] = 1 : d(v, w) \leq t$$

where $d(v, w)$ is the shortest path distance between v and w. Moreover, denote $v.r[i] \in \{0, 1\}^p$ as a binary indicator of whether v can reach a location of service type i within t minutes, such that

$$v.r[i] = 1 \iff \exists w \in V, w.l[i] = 1 : d(v, w) \leq t$$

then, we have

$$\forall v \in R, v.r = \mathbf{1}$$

Taking inspriation from the approach used by Barbieri et al. ([BDMM23], 2.1.1), the modified algorithm will use a modified graph search algorithm to find the 15-Minute City with the service locations of each type as the source nodes, rather than searching from every vertex in the graph. This approach is expected to be more efficient as the number of service locations is expected to be far smaller than the number of vertices in the graph.

Furthermore, a number of other literature works studied in Chapter 2 have mentioned the use of a graph search algorithm to find the 15-Minute City or the travelling time from the source location of interest. Notably, Caselli et al. (Section 2.1.2, [CCRZ22]) defined 15 Minute City from the "neighbour cores" and Rhoads et al. (Section 2.1.3, [RSRBH23]) used a modified Dijkstra's algorithm to comupte a 15 Minute City on side-walk networks with a walkability analysis. The grid tessellatin approaches used by Gaglione et al. (Section 2.2.1, [GGZC22]) and Olivari et al. (Section 2.2.2, [OCNG23]) search for travelling time within a map.

The solution proposed in this thesis is designed to be more general and adaptable to different types of graphs and service locations. Therefore, the solution to the problem stated in this section can be considered as an adoption or a modifictaion to these approaches to the 15-Minute City problem, specifically with an improvment of the computational efficieny in mind. The next section will discuss the proposed solutions

Chapter 4

Proposed Solutions

4.1 Dijkstra's Algorithm

The first solution proposed in this thesis is a modified version of Dijkstra's algorithm, which is a well-known algorithm for finding the shortest path between two vertices in a weighted graph. The algorithm works by selecting the node with the smallest distance from the source node and updating the distance of its neighbors. This process is repeated until all vertices have been visited. The algorithm is efficient for finding the shortest path between two vertices in a graph with non-negative edge weights. However, it is not suitable for graphs with negative edge weights.

Adapting the methodogies used by Pezzica et al. [CLRS22], which incooperate the Dijkstra's algorithm with a (Minimum) Priority Queue data structure. The data structure maintains a dynamic set S of elements, each set element in S has a key and it supports the following dynamic-set operations.

- INSERT$(S; x; k)$: inserts element x with key k into set S.
- MINIMUM(S): returns element of S with smallest key.
- EXTRACT-MIN(S): removes and returns element of S with smallest key.
- DECREASE-KEY$(S; x; k)$: decreases value of element x's key to k. Assumes $k \leq x$'s current key value.

All operations take $O(\log n)$ time in an n-element heap with the exception of MINIMUM(S) being $\Theta(1)$.

We extend the algorithm so that the algotithm will stop once all nodes within t minutes have been visited. For each node considered in the algorithm, a new label $v.d$ is created where $v.d \in \mathbb{R}_{\geq 0}$ representing the distance from the current source node of the algorithm, initialised to ∞.

As the 15-Minute City concept is primarily used to study cities' characteristics, the input graph of the algorithm is expected to be far larger than each 15 minute area. Therefore, it is necessary to stop the algorithm once all nodes within t minutes have been visited to prevent the algorithm from running indefinitely. The modified algorithm is shown in Algorithm 1.

Algorithm 1 Modified Dijkstra's Algorithm

Input: A graph $G(V, E)$, weights $w : E \to \mathbb{R}_{\geq 0}$, source vertex s,
time threshold t and i denotes the index of the service type
Output Assign $v.r[i] = 1$ for vertices that can reach to source node s within threshold t

```
for each vertex v ∈ V do
    v.d ← ∞
end for
s.d ← 0
Q ← ∅
for each vertex v ∈ V do
    INSERT(Q, v)
end for
while Q ≠ ∅ do
    v ←EXTRACT-MIN(Q)
    if v.d > t then
        Q ← ∅                                   ▷ Break out of While loop
    else
        v.r[i] ← 1
        for each vertex u ∈ Adj[v] do
            if u.d > v.d + w(u, v) then
                u.d ← v.d + w(u, v)
                DECREASE-KEY(Q, u, u.d)
            end if
        end for
    end if
end while
```

The modified Dijkstra's algorithm shown above only searches for vertices within t minutes from a single source node. For our context of the 15-Minute City, this needs to run for each location of each service type. The full algorithm as the solution of the problem is shown in Algorithm 2.

Algorithm 2 15-Minute City Algorithm

Input: A graph $G(V, E)$, weights $w : E \to \mathbb{R}_{\geq 0}$, a time threshold t
and a list S of service vertices of p types
Output Set $R \subseteq V$ representing the t-Minute City

```
for all vertex v ∈ V do
    v.r ← {0}^p
    v.l ← {0}^p
end for
for all service v ∈ S do
    v.l[i] ← 1 for each type i of service v contains
end for
for each service type i ∈ {1, ..., p} do
    for each vertex s where s.l[i] = 1 do
        Modified_Dijkstra(G, w, s, t, i)
    end for
end for
R ← ∅
for each vertex v ∈ V do
    if v.r = 1 then
        R ← R ∪ {v}
    end if
end for
```

4.1.1 Analysis

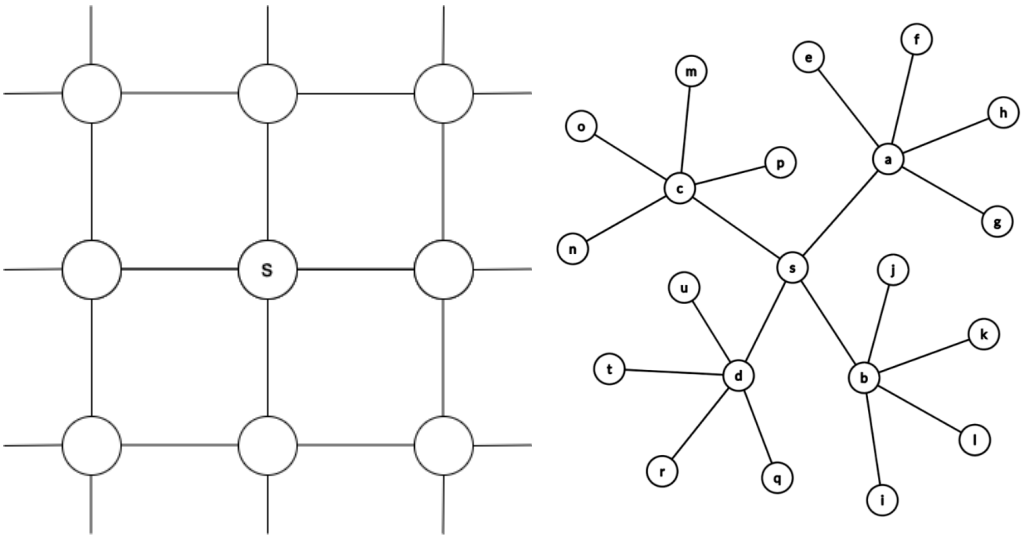
The time complexity of the modified Dijkstra's algorithm depends on the following:

- Initialisation: $O(V)$
- INSERT: $|V| \cdot O(|\text{INSERT}|) = O(|V| \cdot |\text{INSERT}|)$
- EXTRACT-MIN: $|V| \cdot O(|\text{EXTRACT-MIN}|) = O(|V| \cdot |\text{EXTRACT-MIN}|)$
- DECREASE-KEY: $|E| \cdot O(|\text{DECREASE-KEY}|) = O(|E| \cdot |\text{DECREASE-KEY}|)$

The time complexity of the algorithm is also affected by the data structure used to implement the priority queue. A binary heap is a common choice for implementing a priority queue, which has a time complexity of $O(\log V)$ for INSERT, EXTRACT-MIN and DECREASE-KEY operations. However, if a Fibonacci heap is used instead, the time complexity of the operations is reduced to $\Theta(1)$, $O(\log V)$ and $\Theta(1)$ respectively.

As the latter two operations in the algorithm dominates the former two operations, the time complexity of algorithm 1 is $O((V + E) \log V)$ if a binary heap is implemented. This can be reduced to $O(V \log V + E)$ if a Fibonacci heap is considered instead.

For the complete 15-Minute City Algorithm 2, the algorithm is run for each location of each service type. Denote q as the maximum number of locations for any service type, the time complexity of the algorithm is $O(p \cdot q \cdot (V \log V + E))$ if a binary heap is implemented and $O(p \cdot q \cdot (V \log V + E))$ if a Fibonacci heap is implemented. In both cases, the time complexity consider the size of the entire graph, which could be arbitrarily large when a city or a large area is studied. Due to the fact that the modified dijsktra's algorithm stops once all nodes within weight t are searched, it is important to note that the actual complexity of the algorithm could be potentially much smaller.



(a) Example of a grid city

(b) Example of a tree city at 2 levels

Cases where the city is larger than the "15-Minute City"

Example Consider a city with only square grids and each edge has a weight of 1 (see image 4.1a). The algorithm will effectively search a total of $1,024$ edges and 545 nodes for $t = 15$.

In general, for a grid city (where each node has a degree of 4) given a time threshold t, the number of nodes and edges searched by the modified dijastra algorithm can be calculated as follows:

$$\begin{aligned} |\text{Edges}| &= (2 \cdot (t+1))^2 \\ |\text{Vertices}| &= 1 + 2 \cdot (t+1) \cdot (t+2) \end{aligned}$$

This structure of a grid-like city can be applicable to cities such as New York and Barcelona.

Generalisation However, if the graph of interest has the following characteristics: starting from the source node s, every node has d successors with d edges of weight 1. (An illustration of the graph with 2 levels and $d = 4$ are shown in figure 4.1b) In this arrangment, given a time threshold t and d the number of nodes branching out from each parent node, the number of nodes and edges need to be visited are as follows:

$$\begin{aligned} |\text{Edges}| &= \sum_{l=1}^{t+1} d^l = d \cdot \left(\frac{1 - d^{t+1}}{1 - d} \right) \\ |\text{Vertices}| &= 1 + |\text{Edges}| \end{aligned}$$

For example, setting $t = 15$ and $d = 4$, the graph will have $|E| = 5,726,623,060$ and $|V| = 5,726,623,061$. The time complexity of the modified Dijsktra's algorithm is then:

$$\begin{aligned} O(V) &= O(\sum_{l=1}^{t+1} d^l) = O(d^{t+1}) \\ O(E) &= O(1 + \sum_{l=1}^{t+1} d^l) = O(d^{t+1}) \end{aligned}$$

and

$$O(V \log V + E) = O(d^{t+1} \log d^{t+1} + d^{t+1}) = O(d^{t+1} \log d^{t+1})$$

Therefore, the time complexity of the 15-Minute City algorithm on this graph is:

$$O\left(n \cdot m \cdot d^{t+1} \log d^{t+1}\right)$$

This can be considered as the worst case in time complexity for the algorithm with the maximum possible of degree d and time threshold t.

To generalise this notation to edge weights other than 1, define ϵ as the minimum edge weight in the graph. Then the algorithm can travel at most $\lfloor t/\epsilon \rfloor$ edges in t minutes. Therefore, the time complexity of the algorithm can be expressed as:

$$O\left(n \cdot m \cdot d^{1+\lfloor t/\epsilon \rfloor} \log d^{1+\lfloor t/\epsilon \rfloor}\right)$$

Cases where the city is smaller than the "15-Minute City"

In this case, the city is smaller than the 15-Minute City, the algorithm will search all nodes in the graph. The time complexity of the algorithm is simply just

$$O(V \log V + E)$$

4.2 Uniform Cost Search Adaption

For the space complexity of the propsed algorithm above, it is important to note that the modification algorithm of Dijkstra's algorithm inserts all V vertices of the graph into the priority queue set Q, this is repeated for each service location of each type. Therefore, the space complexity would be $O(n \cdot m \cdot V)$. Hence the algorithm proposed may not be suitable for large graphs for space complexity.

The problem described here can be solved by adapting a so-called "Uniform Cost Search" algorithm (also known as "Dijkstra for large graphs"). The algorithm is an extension of Best-first search and it is similar to Dijkstra's algorithm. However, Uniform Cost Search algorithm does not insert all vertices into the priority queue. Instead, it only inserts the vertices that are reachable within the time threshold t. A modification of this algorithm is shown in Algorithm 3. This algorithm can then be used in Algorithm 2 to replace the modified Dijkstra's algorithm in 1.

Algorithm 3 Modified Dijkstra's Algorith 2

Input: A graph $G(V, E)$, weights $w : E \to \mathbb{R}_{\geq 0}$, source vertex s,
time threshold t and i denotes the index of the service type
Output Assign $v.r[i] = 1$ for vertices that can reach to source node s within threshold t

```
Q ← ∅                                          ▷ Initialise an empty priority queue
INSERT(Q, s)
while Q ≠ ∅ do
    v ←EXTRACT-MIN(Q)
    if v.d > t then
        Q ← ∅                                  ▷ Break out of While loop
    else
        v.r[i] ← 1
        for each vertex u ∈ Adj[v] do
            if u ∉ Q then
                u.d ← v.d + w(u, v)
                INSERT(Q, u)
            else if u.d > v.d + w(u, v) then
                u.d ← v.d + w(u, v)
                DECREASE-KEY(Q, u, u.d)
            end if
        end for
    end if
end while
```

4.2.1 Analysis

In this implementation, the time complexity remain the same as Dijkstra's algorithm in the worst case scenario. However, the algorithm is more efficient in practice as it only inserts

vertices that are reachable within the time threshold t into the priority queue. Here, the space complexity becomes

$$O\left(n \cdot m \cdot d^{1+\lfloor t/\epsilon \rfloor}\right)$$

or

$$O\left(n \cdot m \cdot V\right)$$

if the city is smaller than the 15-Minute City.

4.3 Inspiration from Johnson's algorithm

Johnson's algorithm is an all-pairs shortest path algorithm to find shortest paths between every pair of nodes in a graph. It uses both Dijkstra and Bellman-Ford as subroutines and performs better than Floyd-Warshall algorithm in sparse graphs. For the goal of the 15-Minute City algorithm, the set of service vertices can be far smaller than the entire graph in size. Thus, all-pairs shortest path algorithms are not optimal solutions to the problem. However, Johnson's algorithm's approach in adjusting the weights to make all edges non-negative can be applied to the 15-Minute City problem. The algorithm works by adding a new vertex s to the graph and adding edges from the new vertex to all vertices of the graph, before applying Bellman-Ford algorithm.

This approach can be applied to the 15-Minute City problem by adding a new vertex s to the graph and adding edges from the new vertex to all service vertices of the same type. This allows us to eliminate the inner loop of Algorithm 2 where Modified_Dijkstra(G, w, s, t, i) is called. This alternate approach of the 15 Minute City algorithm is shown in Algorithm 4.

Algorithm 4 15-Minute City Algorithm 2

Input: A graph $G(V, E)$, weights $w : E \to \mathbb{R}_{\geq 0}$, a time threshold t and a list S of service vertices of p types
Output Set $R \subseteq V$ representing the t-Minute City

```
for all vertex v ∈ V do
    v.r ← {0}^n
    v.l ← {0}^n
end for
for all service v ∈ S do
    v.l[i] ← 1 for each type i of service v contains
end for
for each service type i ∈ {1, ..., n} do
    Create a new vertex s
    Add edges from s to all vertices v where v.l = i and w(s, v) = 0
    Modified_Dijkstra_2(G, w, s, t, i)
    Remove s and all edges connected to it
end for
R ← ∅
for each vertex v ∈ V do
    if v.r = 1 then
        R ← R ∪ {v}
    end if
end for
```

4.3.1 Analysis

For each service type, the proposed approach increases the number of vertices by 1 and the number of edges by the number of service vertices of the same type. Denote q_w as the maximum number of service vertices of the same type, then for each service type, Modified_Dijkstra_2 algorithm (Algorithm 3) is run, where the algorithm starts from the newly inserted vertex s, it visits at most q_w vertices and continues its search as before. The time complexity of the Modified_Dijkstra_2 algorith is then:

$$O(q_w \cdot (d^{1+\lfloor t/\epsilon \rfloor} \log d^{1+\lfloor t/\epsilon \rfloor})) = O((d^{1+\lfloor t/\epsilon \rfloor} \log d^{1+\lfloor t/\epsilon \rfloor}))$$

and space complexity:

$$O(q_w \cdot d^{1+\lfloor t/\epsilon \rfloor}) = O(d^{1+\lfloor t/\epsilon \rfloor})$$

which are the same as before.

However, the time and space complexity of the 15-Minute City algorithm are now:

$$O\left(p \cdot d^{1+\lfloor t/\epsilon \rfloor} \log d^{1+\lfloor t/\epsilon \rfloor}\right) \text{ and } O\left(p \cdot d^{1+\lfloor t/\epsilon \rfloor}\right)$$

respectively, which are smaller by an order of q_w compared to the previous algorithm.

4.4 Remove duplicate search

Although Algorithm 4 has improved on efficiency compared to the original proposed algorithm (Algorithm 2). There is still room for improvment for cases where more than 1 service type is located at the same vertex. In this case, the same set of nodes will be searched multiple times for each service type. The below algorithm (5 and 6) shows a further modification to Algorithm 4 to remove duplicate search.

Algorithm 5 15-Minute City Algorithm 3

Input: A graph $G(V, E)$, weights $w : E \to \mathbb{R}_{\geq 0}$, a time threshold t and a list S of service vertices of p types

Output Set $R \subseteq V$ representing the t-Minute City

```
for all vertex v ∈ V do
    v.r ← {0}^n
    v.l ← {0}^n
end for
for all service v ∈ S do
    v.l[i] ← 1 for each type i of service v contains
    v.k ← 0                                  ▷ Indicate whether the vertex has been searched
end for
for each service type i ∈ {1, ..., n} do
    Create a new vertex s and denote a set S = {v : i ∈ v.l and v.k = 0}
    Add edges from s to all vertices in S and initialise w(s, v) = 0
    Modified_Dijkstra_2(G, w, s, S, t)
    Remove s and all edges connected to it
end for
R ← ∅
for each vertex v ∈ V do
    if v.r = 1 then
        R ← R ∪ {v}
    end if
end for
```

Algorithm 6 Modified Dijkstra's Algorith 3

Input: A graph $G(V, E)$, weights $w : E \to \mathbb{R}_{\geq 0}$, source vertex s,
S set of vertices of type i and time threshold t

Output Assign $v.r[i] = 1$ for vertices that can reach to source node s within threshold t, for each service type i where $v.l[i] = 1$

```
Q ← ∅                                          ▷ Initialise an empty priority queue
INSERT(Q, s)
while Q ≠ ∅ do
    v ←EXTRACT-MIN(Q)
    v.k ← 1                                    ▷ Mark the vertex as searched
    if v ∉ S then
        w ← v.parent
        v.l ← w.l
    end if
    I ← v.l                                    ▷ Get the set of service types v contains
    if v.d > t then
        Q ← ∅                                  ▷ Break out of While loop
    else
        for each service type i ∈ I do
            v.r[i] ← 1
        end for
        for each vertex u ∈ Adj[v] do
            u.parent ← v
            if u ∉ Q then
                u.d ← v.d + w(u, v)
                INSERT(Q, u)
            else if u.d > v.d + w(u, v) then
                u.d ← v.d + w(u, v)
                DECREASE-KEY(Q, u, u.d)
            end if
        end for
    end if
end while
```

Note: The implementation represented here is not safe. See notes.

Bibliography

[ABCM22] Z. Allam, S. E. Bibri, D. Chabaud, and C. Moreno, "The Theoretical, Practical, and Technological Foundations of the 15-Minute City Model: Proximity and Its Environmental, Social and Economic Benefits for Sustainability," *Energies*, vol. 15, no. 16, p. 6042, Aug. 2022. [Online]. Available: https://www.mdpi.com/1996-1073/15/16/6042

[BDMM23] L. Barbieri, R. D'Autilia, P. Marrone, and I. Montella, "Graph Representation of the 15-Minute City: A Comparison between Rome, London, and Paris," *Sustainability*, vol. 15, no. 4, p. 3772, Feb. 2023. [Online]. Available: https://www.mdpi.com/2071-1050/15/4/3772

[BRSR+21] C. Bustos, D. Rhoads, A. Solé-Ribalta, D. Masip, A. Arenas, A. Lapedriza, and J. Borge-Holthoefer, "Explainable, automated urban interventions to improve pedestrian and vehicle safety," *Transportation Research Part C: Emerging Technologies*, vol. 125, p. 103018, Apr. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0968090X21000498

[BW17] E. Bosina and U. Weidmann, "Estimating pedestrian speed using aggregated literature data," *Physica A: Statistical Mechanics and its Applications*, vol. 468, pp. 1–29, Feb. 2017. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0378437116306604

[CCRZ22] B. Caselli, M. Carra, S. Rossetti, and M. Zazzi, "Exploring the 15-minute neighbourhoods. An evaluation based on the walkability performance to public facilities," *Transportation Research Procedia*, vol. 60, pp. 346–353, Jan. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352146521009455

[CDM11] L. J. Carr, S. I. Dunsiger, and B. H. Marcus, "Validation of Walk Score for estimating access to walkable amenities," *British Journal of Sports Medicine*, vol. 45, no. 14, pp. 1144–1148, Nov. 2011. [Online]. Available: https://bjsm.bmj.com/lookup/doi/10.1136/bjsm.2009.069609

[CDSKL19] D. Capasso Da Silva, D. A. King, and S. Lemar, "Accessibility in Practice: 20-Minute City as a Sustainability Planning Goal," *Sustainability*, vol. 12, no. 1, p. 129, Dec. 2019. [Online]. Available: https://www.mdpi.com/2071-1050/12/1/129

[CLRS22] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms, fourth edition.* MIT Press, 2022. [Online]. Available: https://books.google.it/books?id=HOJyzgEACAAJ

[CLZ20] Y. Chai, C. Li, and Y. Zhang, "A new time-geography research framework of community life circle," *Progress in Geography*, vol. 39, no. 12, pp. 1961–1971, 2020. [Online]. Available: http://www.progressingeography.com/CN/10.18306/dlkxjz.2020.12.001

[Dij59] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, pp. 269–271, Dec. 1959. [Online]. Available: https://doi.org/10.1007/BF01386390

[GGZC22] F. Gaglione, C. Gargiulo, F. Zucaro, and C. Cottrill, "Urban accessibility in a 15-minute city: a measure in the city of Naples, Italy," *Transportation Research Procedia*, vol. 60, pp. 378–385, 2022. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2352146521009509

[igr] "igraph," http://igraph.org/, accessed: 2024-03-06.

[KGSS23] A. R. Khavarian-Garmsir, A. Sharifi, and A. Sadeghi, "The 15-minute city: Urban planning and design efforts toward creating sustainable neighborhoods," *Cities*, vol. 132, p. 104101, Jan. 2023. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0264275122005406

[LBD22] F. T. Lima, N. C. Brown, and J. P. Duarte, "A Grammar-Based Optimization Approach for Designing Urban Fabrics and Locating Amenities for 15-Minute Cities," *Buildings*, vol. 12, no. 8, p. 1157, Aug. 2022. [Online]. Available: https://www.mdpi.com/2075-5309/12/8/1157

[LC23] F. T. Lima and F. Costa, "The Quest for Proximity: A Systematic Review of Computational Approaches towards 15-Minute Cities," *Architecture*, vol. 3, no. 3, pp. 393–409, Jul. 2023. [Online]. Available: https://www.mdpi.com/2673-8945/3/3/21

[MAC+21] C. Moreno, Z. Allam, D. Chabaud, C. Gall, and F. Pratlong, "Introducing the "15-Minute City": Sustainability, Resilience and Place Identity in Future Post-Pandemic Cities," *Smart Cities*, vol. 4, no. 1, pp. 93–111, Jan. 2021. [Online]. Available: https://www.mdpi.com/2624-6511/4/1/6

[MB22] E. Marchigiani and B. Bonfantini, "Urban Transition and the Return of Neighbourhood Planning. Questioning the Proximity Syndrome and the 15-Minute City," *Sustainability*, vol. 14, no. 9, p. 5468, May 2022. [Online]. Available: https://www.mdpi.com/2071-1050/14/9/5468

[OCNG23] B. Olivari, P. Cipriano, M. Napolitano, and L. Giovannini, "Are Italian cities already 15-minute? Presenting the Next Proximity Index: A novel and scalable way to measure it, based on open data," *Journal of Urban Mobility*, vol. 4, p. 100057, Dec. 2023. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2667091723000134

[PAMC24] C. Pezzica, D. Altafini, F. Mara, and C. Chioni, "Travel-time in a grid: Modelling movement dynamics in the "minute city"," in *Innovation in Urban and Regional Planning*, A. Marucci, F. Zullo, L. Fiorini, and L. Saganeiti, Eds. Cham: Springer Nature Switzerland, 2024, pp. 657–668. [Online]. Available: https://doi.org/10.1007/978-3-031-54118-6_58

[RSRBH23] D. Rhoads, A. Solé-Ribalta, and J. Borge-Holthoefer, "The inclusive 15-minute city: Walkability analysis with sidewalk networks," *Computers, Environment and Urban Systems*, vol. 100, p. 101936, Mar. 2023. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0198971522001806

[SWL20] S. Sarkar, H. Wu, and D. M. Levinson, "Measuring polycentricity via network flows, spatial interaction and percolation," *Urban Studies*, vol. 57, no. 12, pp. 2402–2422, Sep. 2020. [Online]. Available: http://journals.sagepub.com/doi/10.1177/0042098019832517

[Wal] "Walk score," https://www.walkscore.com/methodology.shtml, accessed: 2024-03-06.

[WDL+19] M. Weng, N. Ding, J. Li, X. Jin, H. Xiao, Z. He, and S. Su, "The 15-Minute Walkable Neighborhoods: Measurement, Social Inequalities and Implications for Building Healthy Communities in Urban China," *Journal of Transport & Health*, vol. 13, pp. 259–273, Jun. 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2214140518305103

[ZZK+23] S. Zhang, F. Zhen, Y. Kong, T. Lobsang, and S. Zou, "Towards a 15-minute city: A network-based evaluation framework," *Environment and Planning B: Urban Analytics and City Science*, vol. 50, no. 2, pp. 500–514, Feb. 2023. [Online]. Available: http://journals.sagepub.com/doi/10.1177/23998083221118570