Detailed Project Plan

0 Project Information

Project title: Theoretical Research on Online Matching (Gruop 1)

Project supervisor: Zhiyi Huang [1]

Group members: KinHei Kwok [2]

1 Introduction

Matching algorithms, or more specifically referred as **Graph Matching algorithms** under the context of Traditional Computer Science (TCS), has been a fundamental part of Graph theory due to its wide applications (Huang et al., 2018a). For its popularity in the TCS scene, readers are assumed to have a basic understanding on the problem definition of online graph matching, and those who are unfamiliar with the literature are suggested to go through the Appendix **??** for a gentle introduction to the topic. While the topic has been around the scene ever since the founding of the Graph theory field and solutions of the original problem (Dénes, 1931; Philip, 1935) have been widely circulated, there are still many open problems to be answered for the *online* variant of the problem, and the field remains to be under active development.

1.1 Current Research Frontier

The following topics of particular significance are shortlisted for readers to better understand the current focus of the HKU TCS Group regarding graph matching.

Online Bipartite Matching

This subclass of online matching problem restricts the input graph to be an instance of *bipartite graph*, denoted as $\mathrm{G}((,\mathrm{L}),\mathrm{R},\mathrm{E})$, where $\mathrm{L}\cup\mathrm{R}$ is the set of vertices, with L and R representing the offline and online vertices respectively. Under this setting, while all vertices $\mathrm{L}\cup\mathrm{R}$ are revealed to the algorithm on initialization, all edges $(u\in\mathrm{L}, v\in\mathrm{R})$ are only known to the algorithm on arrival of v, i.e. an online update indicating v's availability is received. Moreover, the algorithm is required to either irreversibly match it to some unmatched neighbor u (i.e. u will not be available for future arrivals of $v'\in R$), or to leave v unmatched forever. Since the algorithm is forced to make decisions without full knowledge of the graph, it is doomed to make sub-optimal decisions compared to its offline counterparts, and its performance is evaluated by comparing the output against the maximum matching under

[1]Department of Computer Science, The University of Hong Kong, Email: Zhiyi@cs.hku.hk
[2]Department of Computer Science, The University of Hong Kong, Email: haleyk@hku.hk

some benchmark, with the most trivial examples being **Maximum cardinality bipartite matching** and **Maximum weighted bipartite matching**. The choice of benchmark has gave rise to various open problems. Works in the scene could usually be classified into one of the following categories:

1. **Proposing some optimal solution for some setting.** Notable examples are the **Ranking algorithm** (Karp et al., 1990) and the **Water-level algorithm** (Kalyanasundaram and Pruhs, 2000).
2. **Analyzing extensions of some existing popular algorithms.** Typical works along this line of work extends either the Ranking algorithm or the Water-level algorithm to some other setting which it was not originally designed for, possibly a *de novo* setting proposed by the paper. Papers published by our group usually falls under the group (Huang et al., 2018a).
3. **Improving current analysis.** For most works in the literature, *improvement* here refers to improving the *tightness* of the bound of performances (such as Huang et al. (2018b)), i.e. closing the gap between the best possible and guaranteed performances in the worst case scenario. In some rare occurrences, it may refer to *improving* the *intuitiveness* of the analysis, such as the Randomized Primal-Dual Framework (Devanur et al., 2013).

Randomized Primal-Dual Framework

While the technical details of the framework is out of the scope of this non-technical project plan, a high-level brief introduction is provided in Appendix B for completeness, as it is expected that the framework will be am integral part of our work. This framework presented a generic proof that enabled the community to better understand the characteristics of the Ranking algorithm, and has been foundation stone of most of the proofs in the field for the past few years, such as the Fully Online Matching Model (Huang et al., 2018a).

Fully Online Matching Model

Recently Huang et al. (2018a) has proposed a new model where both L and R are made online. To accommodate this change, each update could signify *either* the arrival or the departure (i.e. deadline) of the node v, and the algorithm need not to finalize its decision w.r.t. v immediately on its arrival, but any time before (or on) receiving its departure update. Note that the original model is a special case of this *de novo* model, where all online vertices' having arrival updates followed by departure updates immediately, and all offline vertices have their departure updates postponed indefinitely.

While the tentative title of this project "Theoretical Research on Online Matching" enjoys an extremely broad scope, it is expected that the work will be mostly related to the fully online (bipartite) matching model, due to the lack of room for improvement in the original model. On the other hand, the fully online matching model provides a lot of exciting opportunities: While other research groups has recognized the potential of this model and published works based on it (Ashlagi et al., 2018; Dutta and Sholley, 2018), it is still less saturated compared to the original model that has been studied for decades. I believe that this topic is a perfect candidate for my Final Year Project, as

- It promises room for work;
- I have studied all the prerequisites of this topic, namely dual LP and graph matching;
- It provides a hands-on experience on a challenging topic to prepare for my post graduate studies in the future;
- It is one of the most well-recognized fields in the TCS community.

1.2 Open Problems

These settings, sorted by my preferences non-decreasingly, has been worked actively before the fully online model was presented, are yet to be solved in the context of the fully online model (and also in the original model). The following settings are not mutually exclusive with each other and could be added on top of the online model simultaneously for analysis.

Online Maximum Weighted Matching: Currently, there is a rather significant discrepancy between the best known bounds of the two cases in the original model (with random arrival order, see the following paragraph). caused by the fact that state-of-the-art of the unweighted analysis relies heavily on symmetry (Karande et al., 2011; Mahdian and Yan, 2011), which the weighted analysis could not rely on (Huang et al., 2018c). A similar branch of work also exists in the context of the fully online model. While the original paper only considered the unweighted setting, recent works by some other research groups has generalized the fully online model to a weighted setting (Ashlagi et al. (2018), Dutta and Sholley (2018)), and an attempt to extend their works would be interesting and valuable.

Random Arrival Order: In the context of **Algorithm complexity analysis**, worst-case analysis is usually used as default. Under the context of graph matching, most of the community's works used to assume the nodes of the online side could show up in arbitrary order, and benchmark the algorithm's performances on the *worst* sequence w.r.t. the algorithm's performance. The field under this assumption has already saturated, as Karp et al. (1990) proposed an optimal solution under the integral setting. Meanwhile, in many practical use cases the underlying algorithm is transparent to the online party (e.g. ride sharing), and it may not be well-justified that the online party's arrival order is adapting aggressively against the algorithm's interest (e.g. University admission on a rolling basis). This has inspired the line of "random arrival" analysis which could be treated as an analogy of average-case analysis. Under this setting, the online party's arrival order is assumed to be shuffled uniformly at random (or a distribution specific to the use case), and the optimal result is yet to be found. Yet the current state-of-the-art result has been promising as it goes beyond the $1 - 1/e$ approximation ratio (Huang et al., 2018c), which was believed to be a natural optimal bound shared by many important Computer Science approximation problems.

General Graph Matching: The general setting has been receiving a lot less attention compared with the bipartite setting due to its complexity. For example, the algorithm for finding the maximum (general) graph matching in the offline variant (Edmonds, 1965) is vastly different and more tedious compared to its bipartite counterpart (Kuhn and Yaw, 1955). Yet, risk taking comes with great opportunities – There are a lot more room for development compared to the saturated bipartite scene. In fact, the original fully online model paper (Huang et al., 2018a) also provided some valuable insights onto the general setting as a starting point.

1.3 Tentative Schedule

It is forecasted that the project comes with a high risk of running into bottle-necks due to its challenging nature, and it is impossible to predict our research progress in the upcoming months, a tentative schedule of the project is proposed as in Table 1. Readers should be reminded that the following schedule is highly susceptible to changes.

1.4 Deliverables

While the course expects the students to implement their projects to submit a tangible deliverable (highlighted with red in Table 1), our success on the topic is not guaranteed even with best efforts committed, due to the project's nature. The mid-progress review on mid January is expected to serve as a sanity check as we will have a better understanding in the problem's hardness, while having sufficient time ahead to turn around if needed. Either way, the project is expected to produce a thesis or a survey within the Online Matching field, which is either ready for publishment if the project is successful, or as a tangible proof of involvement in the project.

1.4.1 Project website

Other than this report, the inception stage also requires the student to submit a project website, which is hosted on Haley Kwok's FYP account.

Table 1: Project timetable

Date	Milestones	Course requirement?
29/09/2019	Deliverables of inception • Detailed project plan • Project web page	Yes
Late Oct / Early Nov	Complete background research • Scan through most of the works in the field • Scrutinize at least 3 of them of particular interest • Narrow down the project scope to a specific topic	No
Semester break	Conduct research based on the specific topic	No
13-17/01/2020	First presentation	Yes
Mid Jan	Mid-progress review • Report on preliminary progress • Decide if the targeted result is achievable • If not, justify the cause of impossibility in interim report and select another topic	No
02/02/2020	Deliverables of Elaboration • Preliminary implementation • Detailed interim report	Yes
Early April	Finalize the research results • Decide if the research result is successful enough for publishment	No
19/04/2020	Deliverables of Construction • Finalized tested implementation • Final report	Yes
20-24/04/2020	Final presentation	Yes
05/05/2020	Project exhibition	Yes
03/06/2020	Project competition	Top only

1.4.2 Software/Technologies used

Academic papers produced by this project should be formatted in LaTeXprofessionally. There are no other strict constraints posed on the other deliverables.

The project's website is developed with the Github Pages utility, using the Jekyll templates that are publicly available to all users.

References

Itai Ashlagi, Maximilien Burq, Patrick Jaillet, and Amin Saberi. Maximizing efficiency in dynamic matching markets. *CoRR*, abs/1803.01285, 2018. URL http://arxiv.org/abs/1803.01285.

Nikhil R. Devanur, Kamal Jain, and Robert D. Kleinberg. Randomized primal-dual analysis of ranking for online bipartite matching. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, pages 101–107, Philadelphia, PA, USA, 2013. Society for Industrial and Applied Mathematics. ISBN 978-1-611972-51-1. URL http://dl.acm.org/citation.cfm?id=2627817.2627824.

Chinmoy Dutta and Chris Sholley. Online matching in a ride-sharing platform. *CoRR*, abs/1806.10327, 2018. URL http://arxiv.org/abs/1806.10327.

Kőnig Dénes. Gráfok és mátrixok. *Matematikai és Fizikai Lapok*, 38, 1931.

Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965. doi: 10.4153/CJM-1965-045-4.

Zhiyi Huang, Ning Kang, Zhihao Gavin Tang, Xiaowei Wu, Yuhao Zhang, and Xue Zhu. How to match when all vertices arrive online. *CoRR*, abs/1802.03905, 2018a. URL http://arxiv.org/abs/1802.03905.

Zhiyi Huang, Binghui Peng, Zhihao Gavin Tang, Runzhou Tao, Xiaowei Wu, and Yuhao Zhang. Tight competitive ratios of classic matching algorithms in the fully online model. *CoRR*, abs/1810.07903, 2018b. URL http://arxiv.org/abs/1810.07903.

Zhiyi Huang, Zhihao Gavin Tang, Xiaowei Wu, and Yuhao Zhang. Online vertex-weighted bipartite matching: Beating 1-1/e with random arrivals. *CoRR*, abs/1804.07458, 2018c. URL http://arxiv.org/abs/1804.07458.

Bala Kalyanasundaram and Kirk Pruhs. An optimal deterministic algorithm for online b-matching. *Theor. Comput. Sci.*, 233:319–325, 2000.

Chinmay Karande, Aranyak Mehta, and Pushkar Tripathi. Online bipartite matching with unknown distributions. pages 587–596, 01 2011. doi: 10.1145/1993636.1993715.

Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An optimal algorithm for on-line bipartite matching. In *STOC*, 1990.

H. W. Kuhn and Bryn Yaw. The hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97, 1955.

Mohammad Mahdian and Qiqi Yan. Online bipartite matching with random arrivals: An approach based on strongly factor-revealing lps. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 597–606, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0691-1. doi: 10.1145/1993636.1993716. URL http://doi.acm.org/10.1145/1993636.1993716.

Hall Philip. On representatives of subsets. *J. London Math. Soc.*, 10:26–30, 1935. URL doi:10.1112/jlms/s1-10.37.26.

A Introduction to Graph Matching for non-specialists

This subsection is devoted to introducing the basics of graph matching algorithms. Throughout this chapter, the "car renting" problem will be proposed to show the model's practicalness, with its variations presented to demonstrate its generality, and to highlight the minor differences between models under different settings.

A.1 Maximum Cardinality Graph Matching

Imagine that you are running a car renting company. The company maintains a list of reservations received from the clients, where each reservation consists of a unit demand client, and a list of cars that could be used to satisfy the client's needs. (Clients with demanding more than one cars may make more than one reservations to satisfy their needs) The company is interested in understanding how to allocate (i.e. **match**) cars to clients to align with the company's objective. In this subsection, it is assumed that the company's objective is to maximum the amount of fulfilled reservation.

For example, consider a use case with three customers and cars pictured in Figure 1. Car #1 has all three clients connected to it, indicating that all three clients' demand could be fulfilled by car #1 (albeit car #1 could only be matched once); Car #2 has no clients connected to it, indicating that no demands could be fulfilled by car #2. An optimal allocation is highlighted in red lines. The red lines indicate one of the optimal scheme that maximizes the amount of fulfilled reservations. Note that the optimal scheme need not to be unique[3].

[3] In this example case, matching car #1 to client #2 instead also fulfills two reservations and hence is optimal.

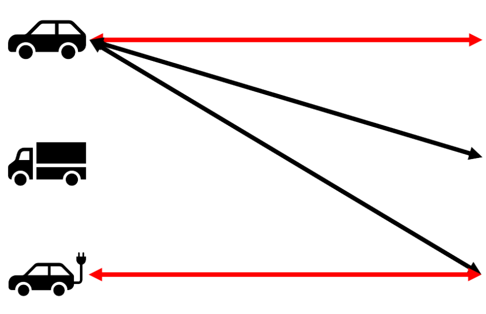

Figure 1: Example use case

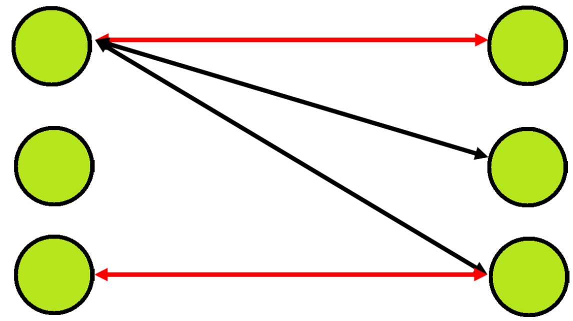

Figure 2: Example graph

To look at this problem from a Computer Scientist's perspective, we may abstract the clients/-cars as a set of vertices V, the reservations as a set of edges connecting the clients and their preferences E. and each problem instance as a simple graph G(V, E). All allocation schemes are abstracted as a matching M which is a subset of E. Motivated by the car renting problem, each car is **matched** to at most one client, and vice versa. This is equivalent as saying "No car/client is involved in more than one allocation", or mathematically written as "No vertex is connected to more than one edge in the matching", formulated as $\forall e, e' \in \mathrm{M} : e \cap e' = \emptyset$.

The objective of the maximum cardinality graph matching problem (often abbreviated to "graph matching" when the context is clear) is to maximize the cardinality of the matching $|\mathrm{M}|$, i.e. the amount of fulfilled reservations.

A.2 Maximum bipartite graph matching

Our car renting use case could be further classified as a bipartite graph matching problem. Bipartite graph matching is a slightly more restrictive class of the (general) graph matching, in which the discussed graph must be bipartite[4], usually represented as $\mathrm{G}(\mathrm{L} \cup \mathrm{R},\ \mathrm{E})$. Thanks to numerous properties guaranteed by the bipartite structure, research progress of this branch is far more advanced than the general setting[5]. Despite of the restriction, the model remains to be relevant in most use cases, such as the car renting example and other client-server applications.

A.3 Maximum weighted graph matching

In reality, each of the reservation need not to generate the same amount of profit. To capture this behavior, the model is extended such that each client v is associated with a budget w_v,

[4]Recall that a graph is bipartite if and only if its nodes could be partitioned into two sets L and R, such that each of the edges connects a node between Land R.

[5]For example, you might have heard of classical theorems that are only applicable to bipartite graphs, such as the Hall's marriage theorem (Philip, 1935) and the Konig's theorem (Dénes, 1931).

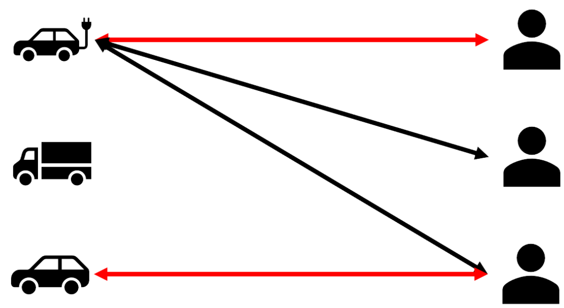

Figure 3: The company doesn't know which car is more preferred by the majority

and the company's objective is altered to maximize its profit instead of fulfilled reservations, i.e. $\sum_{v \in V} w_v$. This modelling could also be extended to the cars, e.g. w_u is a negative number which represents the expense for deploying car u.

A.4 Online bipartite graph matching

Sometimes it may be unrealistic to assume all reservations are known *a priori*, to see this, consider a variant of car renting which provides services on demand - Whenever the client makes a call to the reception, she (a) gets served and pays to the company (i.e. matched to some taxi) or, (b) gets rejected hence doesn't pay (i.e. not matched to any taxi).

To clarify how the model operates, consider an example case which follows from Figure 1. Imagine that the clients came in the order of $3 \to 1 \to 2$. When client #3 approaches the reception, the company could either assign car #1 or car #3 to the client. If the company assigns car #1 to the client, it will not be available for service for client #1 and #2 in the future.

Since the company is forced to make decisions without full disclosure of future demands, e.g. it cannot distinguish between Figure 1 and 3, it may not always generate optimal revenue even if it operates optimally, and such dilemma is the crux of online matching.

We propose the following mathematical abstraction: Denote the graph as $\mathrm{G}(\mathrm{L} \cup \mathrm{R},\ \mathrm{E})$, vertices $v \in \mathrm{L}$ are named as "offline vertices" (The cars on the left in our example), while $u \in \mathrm{R}$ are named as "online vertices" (The clients on the right in our example).

Researchers benchmarks online algorithms' using the notion "approximation ratio" F. Denote $\mathrm{ALG}(\mathrm{G})$ as the matching outputted by the algorithm ALG, and $\mathrm{OPT}(\mathrm{G})$ as the optimal matching (achieved by offline solutions). The approximation ratio is defined as

$$\mathrm{F} \;=\; \min_{G} \left\{ \frac{\mathrm{ALG}(G)}{\mathrm{OPT}(G)} \right\},$$

i.e. the minimal fraction of the optimal matching secured by the algorithm in the worst case scenario.

The first and most famous optimal algorithm is the ranking algorithm proposed by Karp et al. (1990), The work comes with a heart-breaking corollary that "randomization is necessary to achieve the approximation ratio"[6].

The online weighted graph matching problem also has a weighted variant defined similarly to its offline counterpart, yet the most analysis restricts all online nodes to have a zero weight, i.e. $w_v = 0 \ \forall v \in \mathrm{R}$ This limitation may be too restricting to some use cases, as in we cannot assign weights to clients (online vertices) in the car renting problem. To this extent, reader may consider the "fully online" model proposed by Huang et al. (2018a) to overcome this shortcoming, with its weighted variant discussed in the works of Ashlagi et al. (2018); Dutta and Sholley (2018), which is out of this chapter's scope to not overwhelm our readers.

A.5 Fractional graph matching

It was later shown by Kalyanasundaram and Pruhs (2000) that with some slight modifications to the definition of matching, their proposed (determinstic) waterlevel algorithm is optimal.

Consider a variant of the car renting model with a gigantic market. Instead of modelling cars/clients as vertices, garages of B cars/clusters of B clients each are modelled as a single node. This modelling. called b-matching, relaxes the matching constraint in a way that all nodes could be matched for at most B times repetitively. For the special case where $B \to \infty$, called *fractional* matching, (on contrast, the original model is called integral matching) arbitrary *fractions* of vertices may be matched to its neighbour, as long as (a) both nodes along the edge committed the same fraction, and (b) the total fraction committed for any node does not exceed 1.

B Introduction to the Randomized Primal Dual Framework

This section is devoted to provide a gentle introduction to the Randomized Primal Dual Framework for the reader. For simplicity, only the unweighted use case is considered here.

Consider the standard Linear Programming relaxation of the fractional matching problem:

$$
\begin{aligned}
\text{max:} \quad & \sum_{(u,v)\in E} w_v \cdot x_{uv} & \\
\text{s.t.:} \quad & \sum_{v:(u,v)\in E} x_{uv} \le 1 & \forall u \in V \\
& x_{uv} \ge 0 & \forall (u,v) \in E
\end{aligned}
$$

[6] Precisely speaking, the work proved that "no deterministic algorithm achieves an approximation ratio better than 0.5. It is trivial that the naive algorithm which matches the online nodes to any offline node that is unmatched on its arrival achieves the same approximation ratio.

where x_{uv} is the fraction of the edge (u, v) used in the matching.

The program has a corresponding dual LP formulated as follows:

$$\begin{aligned} \text{min: } & \sum_{u \in L \cup R} \alpha_u & \\ \text{s.t.: } & \alpha_u + \alpha_v \geq 1 & \forall (u,v) \in E \\ & \alpha_u \geq 0 & \forall u \in L \cup R \end{aligned}$$

Then, the rest of the framework is dedicated to design a function g. The function is related to the algorithm in the sense that whenever it (randomly) matches the vertices $u \in L$ and $v \in R$ under some parameterized situation λ, it contributes $g(\lambda)$ to α_u and $1 - g(\lambda)$ to α_v. Then, the algorithm is said to be a F-approximation if fulfills the constraint $\alpha_u + \alpha_v \geq 1/F \ \forall (u,v) \in E$, due to the following arguments:

- The algorithm provides a witness for the dual variables α, and its expected performance is ALG, i.e. $\sum_{u \in \mathrm{L} \cup \mathrm{R}} \alpha_u = \mathbf{E}_{\vec{y}}[\mathrm{ALG}]$.
- The witness satisfies the constraint $\alpha_u + \alpha_v \geq 1/F \ \forall (u,v) \in E$, which could be scaled to the original constraints $F(\alpha_u + \alpha_v) \coloneqq \tilde{\alpha_u} + \tilde{\alpha_v} \geq 1 \ \forall (u,v) \in E$.
- The scaled witness is lower-bounded by the optimal fractional solution OPT, i.e. $\mathrm{OPT} \leq \sum_{u \in \mathrm{L} \cup \mathrm{R}} \tilde{\alpha_u}$.

Joining the arguments together gives the following inequality (Huang et al., 2018c):

$$\mathrm{OPT} \ \leq \ \sum_{u \in \mathrm{L} \cup \mathrm{R}} \tilde{\alpha_u} \ = \ \frac{1}{F} \sum_{u \in \mathrm{L} \cup \mathrm{R}} \alpha_u \ = \ \frac{1}{F} \mathop{\mathbf{E}}_{\vec{y}}[\mathrm{ALG}].$$

To give a concrete example, consider the following descriptions of the ranking algorithm under the framework:

- Let y_u be the rank assigned to the vertex $u \in L$ which is drawn from $[0, 1]$ uniformly at random, and $\vec{y}$ as the vectorized ranking of all vertices in L.
- The ranking algorithm matches all vertices $v \in R$ to the available neighbour $u \in L$ with the highest ranking upon arrival, and contributes to dual variables α_u, α_v based on g, λ.
- The function uses the rank y_v as the only parameter, and is implemented as $g(\lambda) = g(y = y_v) = e^{y-1}$. One could show that the dual constraint is satisfied with $F = 1 - 1/e$.

C Outlined official guidelines

The official guideline for detailed project plan supplemented by the course's page is outlined as follows, for the grader's convenience.

- **Project background** The background of the problem has been sufficiently covered in Section 1.1, 1.2 and Appendix **??**
- **Project objective**
  The deliverables of this project is listed in Section 1.4.
- **Project methodology**
  The usage of the Randomized Primal Dual Framework and possibly the Fully Online Model is justified in Section 1.1, and their details are briefly mentioned in Appendix B and Section 1.1 respectively.
- **Project Schedule and Milestones**
  A detailed list of verifiable milestones is listed in Section 1.3.