Principles of approximation algorithms

Outline

Why a performance guarantee?

Approximation algorithms
- Relative performance guarantee
- Additive performance guarantee

Approximation schemes

Typical mistakes

In summary

Learning objectives

At the end of this chapter you will be able to:

- compute the performance ratio, the additive error and the relative error of a heuristic algorithm on an instance, for minimization and maximization problems;
- state when a number is a relative or an additive performance guarantee, and when it is tight;
- prove that a guarantee is tight by exhibiting an instance, or a sequence of instances;
- define an approximation scheme, a PTAS and an FPTAS, and translate the accuracy ε into a performance guarantee.

Prerequisites

- From the module *Heuristic Algorithms*, the chapter «Heuristic algorithms: definitions»: heuristic algorithms, feasible solutions and the value of the solution returned by a heuristic algorithm.
- From the module *Combinatorial Optimization Problems*, the chapters on the combinatorial optimization problems: instances, optimal solutions and the optimal value $z(I)$ of an instance I.

Why a performance guarantee?

Introduction: Why a performance guarantee?

- A heuristic algorithm returns a feasible solution, but how good is it?
- Testing it on some instances tells us little about the next instance
- We look for statements that hold for *every* instance: performance guarantees

Tests and guarantees

- A heuristic algorithm is usually tested on a few instances, and it may behave very differently on the next one.
- A **performance guarantee** is a statement that holds for *every* instance.

A challenge

A heuristic algorithm for a maximization problem returns the value 40 on an instance whose optimal value is 50.

How far is it from the optimum?

Try to give **three different answers** before reading on.

Three answers

The same gap can be measured in three ways:

- the **ratio**
$$\frac{50}{40} = \frac{5}{4};$$
- the **additive** error
$$50 - 40 = 10;$$
- the **relative** error
$$\frac{50 - 40}{50} = \frac{1}{5}.$$

This chapter defines the three measures; later in the course we compute the worst of them over all instances.

Summary

Why a guarantee:

- a few tests say little about the next instance
- a performance guarantee holds for every instance

Three measures of the same gap:

- the ratio, the additive error and the relative error
- for the challenge: $\frac{5}{4}$, 10 and $\frac{1}{5}$

Approximation algorithms

Introduction: Approximation algorithms

- In this section we compare the heuristic value $\mathscr{A}(I)$ with the optimal value $z(I)$, in the worst case over all instances
- We define two kinds of guarantees: **relative** (with ratios) and **additive** (with differences)
- For each of them we see when a guarantee is **tight**, and which is the best possible one

Heuristic value

Definition 1: Heuristic value

Let $\mathscr{A}$ be a ***heuristic algorithm*** *for an* ***optimization problem*** *$\mathscr{P}$. For every* ***instance*** *I of $\mathscr{P}$:*

1. *$z(I)$ is the* ***optimal value*** *of $\mathscr{P}$ on I (assumed positive);*
2. *$\mathscr{A}(I)$ is the* ***heuristic value****, i.e., the objective value of the feasible solution returned by $\mathscr{A}$ on I (assumed positive).*

The feasibility assumption

- We assume that $\mathscr{A}$ always outputs a **feasible solution**.
- Hence, for every instance I,

$$\begin{cases} \mathscr{A}(I) \geq z(I) \text{ and } \dfrac{\mathscr{A}(I)}{z(I)} \geq 1 & \text{if } \mathscr{P} \text{ is a minimization problem,} \\ \mathscr{A}(I) \leq z(I) \text{ and } \dfrac{z(I)}{\mathscr{A}(I)} \geq 1 & \text{if } \mathscr{P} \text{ is a maximization problem.} \end{cases}$$

Measuring the quality

In general, a heuristic algorithm may sometimes generate solutions with an almost optimal value, but in other cases it may produce significantly inferior solutions.

Accordingly, we are interested in "measuring" (whenever possible) the quality of the solutions provided by a heuristic algorithm.

Worst-case behaviour

- We adopt a **worst-case behaviour** point of view.
- We evaluate a heuristic algorithm by looking at the largest possible deviation, over all instances, between the heuristic value and the optimal value.

Two ways to measure the deviation

1. **Relative performance guarantee** (multiplicative guarantee): we measure the quality of the heuristic solution by looking at the *relative error*, that is, how far the heuristic value is from the optimum *in relative terms*, using ratios.

2. **Absolute performance guarantee** (additive guarantee): we measure the quality of the heuristic solution by looking at the *absolute error*, that is, the difference between the heuristic value and the optimal value.

Relative performance guarantee

We start from the **relative** guarantee:
the heuristic value and the optimal value are compared through their **ratio**.

Performance ratio

Definition 2: Performance ratio

The ***performance ratio*** *of* $\mathscr{A}$ *on an instance* I *is*

$$\begin{cases} \dfrac{\mathscr{A}(I)}{z(I)} & \text{if } \mathscr{P} \text{ is a minimization problem,} \\ \dfrac{z(I)}{\mathscr{A}(I)} & \text{if } \mathscr{P} \text{ is a maximization problem.} \end{cases}$$

- Under the feasibility assumption, the performance ratio is at least 1 for every instance I.

α-approximation algorithm

Definition 3: α-approximation algorithm

A heuristic algorithm $\mathscr{A}$ for an optimization problem $\mathscr{P}$ is an ***α-approximation algorithm*** *if there exists a constant $\alpha \in [1, +\infty)$ such that, for every instance I of $\mathscr{P}$,*

$$\begin{cases} \dfrac{\mathscr{A}(I)}{z(I)} \leq \alpha & \text{if } \mathscr{P} \text{ is a minimization problem,} \\ \dfrac{z(I)}{\mathscr{A}(I)} \leq \alpha & \text{if } \mathscr{P} \text{ is a maximization problem.} \end{cases}$$

Any such constant α is called a ***relative performance guarantee*** *of $\mathscr{A}$.*

Tight relative guarantee

- A relative performance guarantee α for $\mathscr{A}$ is **tight** if no smaller constant $\alpha' < \alpha$ is also a relative performance guarantee for $\mathscr{A}$.
- That is, the same inequalities do not hold for all instances with α' in place of α.

How to prove tightness

- Tightness is usually proved by exhibiting an instance I for which the performance ratio attains α, namely

$$\frac{\mathscr{A}(I)}{z(I)} = \alpha \quad \text{(minimization)} \qquad \text{or}$$

$$\frac{z(I)}{\mathscr{A}(I)} = \alpha \quad \text{(maximization)},$$

- or a sequence of instances $\{I_n\}$ such that the corresponding performance ratios get arbitrarily close to α.

Many constants can be relative guarantees of the same algorithm.

Which is the **smallest** one?

Worst-case performance ratio

Definition 4: Worst-case performance ratio

*The **worst-case performance ratio** of $\mathscr{A}$ is, if $\mathscr{P}$ is a minimization problem,*

$$r(\mathscr{A}) = \sup \left\{ \frac{\mathscr{A}(I)}{z(I)} : I \text{ is an instance of } \mathscr{P} \right\},$$

and, if $\mathscr{P}$ is a maximization problem,

$$r(\mathscr{A}) = \sup \left\{ \frac{z(I)}{\mathscr{A}(I)} : I \text{ is an instance of } \mathscr{P} \right\}.$$

Properties of $r(\mathscr{A})$

- The value $r(\mathscr{A})$ is the smallest constant α for which $\mathscr{A}$ is an α-approximation algorithm.
- The relative performance guarantee α is tight if and only if $\alpha = r(\mathscr{A})$.
- If $r(\mathscr{A}) < \infty$, then $\mathscr{A}$ admits a finite relative performance guarantee (equivalently, it has a bounded performance ratio).
- If $r(\mathscr{A}) = \infty$, then $\mathscr{A}$ has no relative worst-case guarantee.

Terminology used in the literature

- A finite relative performance guarantee α is also called an **approximation factor** (or **approximation guarantee**).
- An α-**approximation algorithm** is also called a **constant-factor approximation algorithm**.
- $r(\mathscr{A})$ is also called the **(worst-case) approximation ratio** of $\mathscr{A}$.

Additive performance guarantee

We now repeat the same path with **differences** instead of ratios:
additive error, additive guarantee, tightness and the best guarantee.

Additive error

- The **additive error** (or **absolute error**) of $\mathscr{A}$ on an instance I is

$$\begin{cases} \mathscr{A}(I) - z(I) & \text{if } \mathscr{P} \text{ is a minimization problem,} \\ z(I) - \mathscr{A}(I) & \text{if } \mathscr{P} \text{ is a maximization problem.} \end{cases}$$

- Under the feasibility assumption, this quantity is non-negative for every instance I.

β-additive approximation algorithm

Definition 5: β-additive approximation algorithm

*A heuristic algorithm $\mathscr{A}$ for an optimization problem $\mathscr{P}$ is a β-**additive approximation algorithm** if there exists a constant $\beta \in [0, +\infty)$ such that, for every instance I of $\mathscr{P}$,*

$$\begin{cases} \mathscr{A}(I) - z(I) \leq \beta & \text{if } \mathscr{P} \text{ is a minimization problem,} \\ z(I) - \mathscr{A}(I) \leq \beta & \text{if } \mathscr{P} \text{ is a maximization problem.} \end{cases}$$

*Any such constant β is called an **additive performance guarantee** of $\mathscr{A}$.*

Tight additive guarantee

- An additive performance guarantee β for $\mathscr{A}$ is **tight** if no smaller constant $\beta' < \beta$ is also an additive performance guarantee for $\mathscr{A}$.

- Tightness is usually proved by exhibiting an instance I for which the additive error attains β, namely

$$\mathscr{A}(I) - z(I) = \beta \quad \text{(minimization)} \qquad \text{or}$$

$$z(I) - \mathscr{A}(I) = \beta \quad \text{(maximization)},$$

- or a sequence of instances $\{I_n\}$ such that the corresponding additive errors get arbitrarily close to β.

Worst-case additive error

Definition 6: Worst-case additive error

*The **worst-case additive error** of $\mathscr{A}$ is, if $\mathscr{P}$ is a minimization problem,*

$$e(\mathscr{A}) = \sup\{\mathscr{A}(I) - z(I) : I \text{ instance of } \mathscr{P}\},$$

and, if $\mathscr{P}$ is a maximization problem,

$$e(\mathscr{A}) = \sup\{z(I) - \mathscr{A}(I) : I \text{ instance of } \mathscr{P}\}.$$

Properties of $e(\mathscr{A})$

- The value $e(\mathscr{A})$ is the smallest constant β for which $\mathscr{A}$ is a β-additive approximation algorithm.
- The additive performance guarantee β is tight if and only if $\beta = e(\mathscr{A})$.
- If $e(\mathscr{A}) < \infty$, then $\mathscr{A}$ admits a finite additive performance guarantee (equivalently, it has bounded additive error).
- If $e(\mathscr{A}) = \infty$, then $\mathscr{A}$ has no additive (absolute) worst-case guarantee.

Terminology used in the literature

- A finite additive performance guarantee β is also called an **absolute performance guarantee** or an **additive approximation guarantee**.
- $e(\mathscr{A})$ is also called the **worst-case absolute (additive) error** of $\mathscr{A}$.

Summary (1/2)

Heuristic value:

- $\mathscr{A}(I)$ is the value of the feasible solution returned by $\mathscr{A}$ on I
- it is never better than $z(I)$; we judge $\mathscr{A}$ in the worst case over all instances

Relative guarantee:

- performance ratio $\mathscr{A}(I)/z(I)$ (minimization) or $z(I)/\mathscr{A}(I)$ (maximization), always at least 1
- α-approximation algorithm: ratio at most α on every instance
- the smallest such α is $r(\mathscr{A})$, and α is tight iff $\alpha = r(\mathscr{A})$

Summary (2/2)

Additive guarantee:

- additive error $\mathscr{A}(I) - z(I)$ (minimization) or $z(I) - \mathscr{A}(I)$ (maximization), always non-negative
- β-additive approximation algorithm: additive error at most β on every instance
- the smallest such β is $e(\mathscr{A})$, and β is tight iff $\beta = e(\mathscr{A})$

Tightness:

- shown by an instance that attains the guarantee, or by a sequence of instances that gets arbitrarily close to it

Approximation schemes

Introduction: Approximation schemes

- In this section we measure the quality of a heuristic solution with the **relative error**
- An approximation scheme is a family of algorithms: one for every allowed error ε
- We translate ε into a performance guarantee α, and we classify the schemes by their running time: **PTAS** and **FPTAS**

Relative error

- The **relative error** (also called **percentage error**) of $\mathscr{A}$ on an instance I is

$$\begin{cases} \dfrac{\mathscr{A}(I) - z(I)}{z(I)} & \text{if } \mathscr{P} \text{ is a minimization problem,} \\ \dfrac{z(I) - \mathscr{A}(I)}{z(I)} & \text{if } \mathscr{P} \text{ is a maximization problem.} \end{cases}$$

- Under the feasibility assumption, this quantity is non-negative for every instance I, and it is equal to 0 if and only if $\mathscr{A}(I) = z(I)$.

Relative error and gap

- The relative error divides by the optimal value $z(I)$.
- It must not be confused with the **gap** of the chapter «Combinatorial optimization: definitions» of the module *Combinatorial Optimization Problems*, which divides by the lower bound.
- The gap is never smaller than the relative error (chapter «Heuristic algorithms: definitions» of the module *Heuristic Algorithms*).

Approximation scheme

Definition 7: Approximation scheme

*An **approximation scheme** for an optimization problem $\mathscr{P}$ is a family of heuristic algorithms $\{\mathscr{A}_\varepsilon\}_{\varepsilon\in(0,1)}$ such that, for every $\varepsilon \in (0,1)$ and for every instance I of $\mathscr{P}$,*

$$\begin{cases} \dfrac{\mathscr{A}_\varepsilon(I) - z(I)}{z(I)} \leq \varepsilon & \text{if } \mathscr{P} \text{ is a minimization problem,} \\ \dfrac{z(I) - \mathscr{A}_\varepsilon(I)}{z(I)} \leq \varepsilon & \text{if } \mathscr{P} \text{ is a maximization problem.} \end{cases}$$

Each algorithm $\mathscr{A}_\varepsilon$ of a scheme has relative error at most ε.

Which **performance guarantee** α does this give?

The answer depends on the direction of the optimization.

From ε to α: minimization problems

- For minimization problems,

$$\frac{\mathscr{A}_\varepsilon(I) - z(I)}{z(I)} \leq \varepsilon \qquad \text{for every instance } I \text{ of } \mathscr{P}$$

implies

$$\mathscr{A}_\varepsilon(I) - z(I) \leq \varepsilon\, z(I) \quad \Longleftrightarrow \quad \mathscr{A}_\varepsilon(I) \leq (1+\varepsilon)\, z(I).$$

- Dividing by $z(I) > 0$,

$$\frac{\mathscr{A}_\varepsilon(I)}{z(I)} \leq 1 + \varepsilon \qquad \text{for every instance } I \text{ of } \mathscr{P}.$$

- Hence, $\mathscr{A}_\varepsilon$ is a $(1+\varepsilon)$-approximation algorithm.

From ε to α: maximization problems

- For maximization problems,

$$\frac{z(I) - \mathscr{A}_\varepsilon(I)}{z(I)} \leq \varepsilon \qquad \text{for every instance } I \text{ of } \mathscr{P}$$

  implies

$$z(I) - \mathscr{A}_\varepsilon(I) \leq \varepsilon\, z(I) \quad \Longleftrightarrow \quad \mathscr{A}_\varepsilon(I) \geq (1-\varepsilon)\, z(I).$$

- Since $0 < \varepsilon < 1$ and $z(I) > 0$, dividing by $(1-\varepsilon)z(I)$ gives

$$\frac{z(I)}{\mathscr{A}_\varepsilon(I)} \leq \frac{1}{1-\varepsilon} \qquad \text{for every instance } I \text{ of } \mathscr{P}.$$

- Hence, $\mathscr{A}_\varepsilon$ is a $\left(\frac{1}{1-\varepsilon}\right)$-approximation algorithm.

Error and running time

It is quite natural that the running time of an approximation scheme increases as ε decreases:

the smaller the allowed error, the higher the required running time.

Accuracy (1/2)

- The quantity $\frac{1}{\varepsilon}$ is often called the **accuracy**.
- Indeed, from the defining inequalities of an approximation scheme we obtain, for a **minimization** problem,

$$\frac{\mathscr{A}_\varepsilon(I) - z(I)}{z(I)} \leq \varepsilon \implies \mathscr{A}_\varepsilon(I) - z(I) \leq \varepsilon\, z(I)$$

$$\implies \frac{z(I)}{\mathscr{A}_\varepsilon(I) - z(I)} \geq \frac{1}{\varepsilon}.$$

Accuracy (2/2)

- For a **maximization** problem,

$$\frac{z(I) - \mathscr{A}_\varepsilon(I)}{z(I)} \leq \varepsilon \implies z(I) - \mathscr{A}_\varepsilon(I) \leq \varepsilon\, z(I)$$

$$\implies \frac{z(I)}{z(I) - \mathscr{A}_\varepsilon(I)} \geq \frac{1}{\varepsilon}.$$

- In the special case where $\mathscr{A}_\varepsilon(I) = z(I)$, the denominators above are zero; in this case we interpret the corresponding fractions as $+\infty$, and the inequalities hold trivially.

Not all approximation schemes are equally useful in practice.

We distinguish them according to how the running time depends on the **input size** and on the **accuracy** $\frac{1}{\varepsilon}$.

PTAS

Definition 8: Polynomial-time approximation scheme (PTAS)

*An approximation scheme $\{\mathscr{A}_\varepsilon\}_{\varepsilon\in(0,1)}$ is a **polynomial-time approximation scheme** if, for every fixed $\varepsilon \in (0,1)$, the running time of $\mathscr{A}_\varepsilon$ is polynomial in the size of the instance.*

- Thus, the running time is polynomial in the input size, but it may grow very fast as ε decreases (i.e., as the required accuracy increases).

FPTAS

Definition 9: Fully polynomial-time approximation scheme (FPTAS)

An approximation scheme $\{\mathscr{A}_\varepsilon\}_{\varepsilon\in(0,1)}$ *is a* ***fully polynomial-time approximation scheme*** *if the running time of* $\mathscr{A}_\varepsilon$ *is polynomial both in the size of the instance and in the accuracy* $\frac{1}{\varepsilon}$.

Summary

Relative error:

- $(\mathscr{A}(I) - z(I))/z(I)$ (minimization) or $(z(I) - \mathscr{A}(I))/z(I)$ (maximization)
- it divides by $z(I)$: the gap is never smaller

Approximation schemes:

- relative error at most ε for every $\varepsilon \in (0, 1)$
- a $(1 + \varepsilon)$-approximation (minimization), a $\frac{1}{1-\varepsilon}$-approximation (maximization)
- PTAS: polynomial in the size for every fixed ε; FPTAS: polynomial in $\frac{1}{\varepsilon}$ too

Typical mistakes

Introduction: Typical mistakes

- We collect the mistakes that are made most often with performance guarantees and approximation schemes
- Each of them comes from a definition of this chapter
- Check them before solving the exercises

Typical mistakes (1/3)

Observation 1: Inverting the ratio

The performance ratio is

$$\frac{\mathscr{A}(I)}{z(I)} \quad \textit{for a minimization problem and}$$

$$\frac{z(I)}{\mathscr{A}(I)} \quad \textit{for a maximization problem:}$$

in both cases it is at least 1.

Typical mistakes (2/3)

Observation 2: Proving a guarantee with examples

An instance on which the ratio equals α shows that no constant below α is a guarantee, but it does not prove that α is one: this needs a proof for every instance.

Observation 3: Calling a guarantee tight without an instance

A guarantee α is tight only if some instance, or a sequence of instances, reaches α in the limit.

Typical mistakes (3/3)

Observation 4: Confusing ε and α

An approximation scheme with relative error ε is a $(1+\varepsilon)$-approximation algorithm for a minimization problem, but a $\frac{1}{1-\varepsilon}$-approximation algorithm for a maximization problem.

Observation 5: PTAS and FPTAS

The running time of a PTAS may grow like $n^{1/\varepsilon}$; only for an FPTAS it is polynomial in $\frac{1}{\varepsilon}$ as well.

Summary

Guarantees:

- the performance ratio is always at least 1: do not invert it
- a guarantee needs a proof for every instance; tightness needs an instance, or a sequence of instances

Approximation schemes:

- ε gives $1+\varepsilon$ (minimization) but $\frac{1}{1-\varepsilon}$ (maximization)
- only an FPTAS is polynomial in $\frac{1}{\varepsilon}$ as well

In summary

Introduction: In summary

- We collect the key ideas of the chapter on approximation algorithms
- The dashboard links the guarantees to the heuristic values on the reference instances

Summary (1/2)

- A relative performance guarantee bounds the ratio $\mathscr{A}(I)/z(I)$ (minimization) or $z(I)/\mathscr{A}(I)$ (maximization) over all instances; an additive guarantee bounds the difference.
- A guarantee is tight if it is reached by an instance, or approached by a sequence of instances; the smallest guarantee is the worst-case ratio $r(\mathscr{A})$, or the worst-case additive error $e(\mathscr{A})$.
- An approximation scheme has relative error at most ε for every $\varepsilon \in (0, 1)$; it is a PTAS if the running time is polynomial in the size of the instance for every fixed ε, and an FPTAS if it is polynomial in $\frac{1}{\varepsilon}$ too.

Summary (2/2)

Dashboard.

- A guarantee bounds the gap between a heuristic solution (a lower bound on $z(I)$ for a maximization problem, an upper bound for a minimization problem) and the optimal value, on every instance.
- The dashboards of the chapters of this part of the course collect, for the reference instances, the values of the heuristic algorithms and their guarantees.
- For the VCP, the MCP and the general TSP no constant guarantee exists, unless $P = NP$.