Otoczka wypukła dla zbioru punktów w przestrzeni dwuwymiarowej

Algorytmy Geometryczne

Remigiusz Babiarz · Jakub Własiewicz

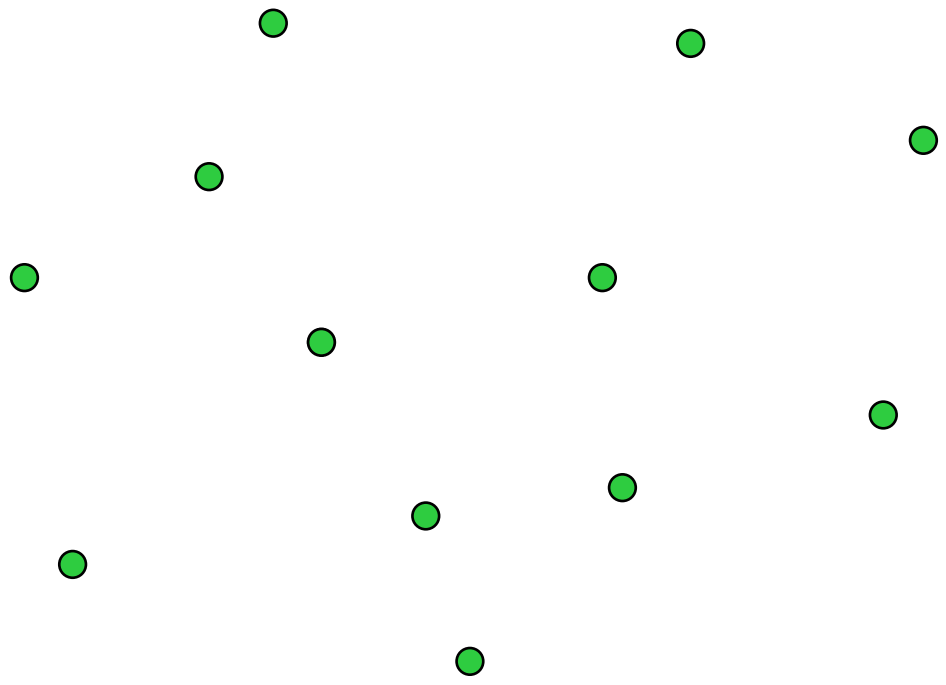
Plan prezentacji

Plan prezentacji

1. Otoczka wypukła

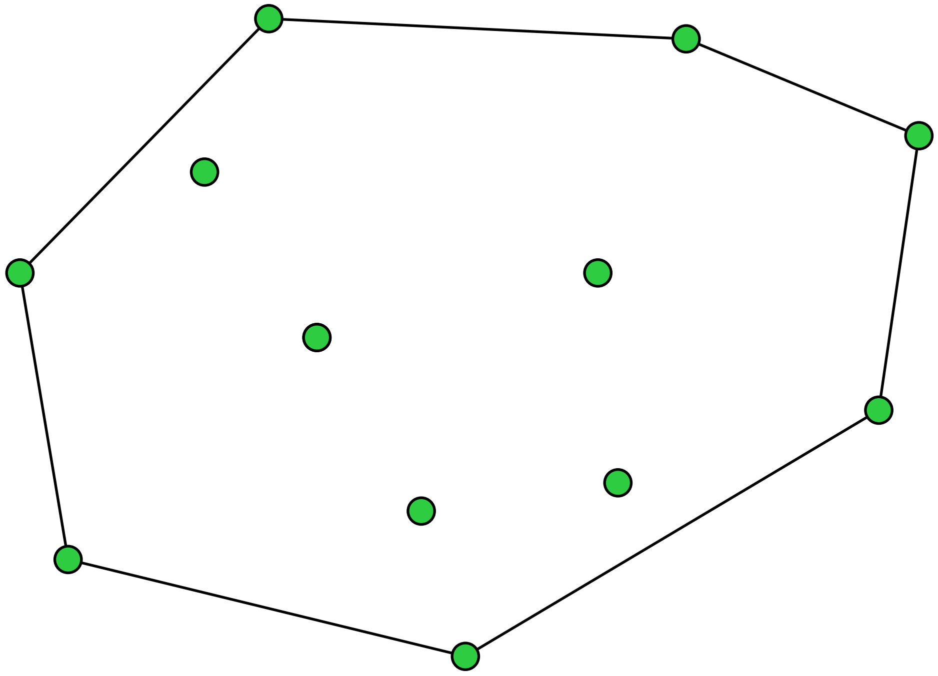
1.1 Definicja

Otoczką wypukłą zbioru punktów nazywamy najmniejszy wielokąt wypukły zawierający ten zbiór. Będziemy oznaczać liczbę punktów zbioru wejściowego jako n, a liczbę punktów otoczki jako h.



1.1 Definicja

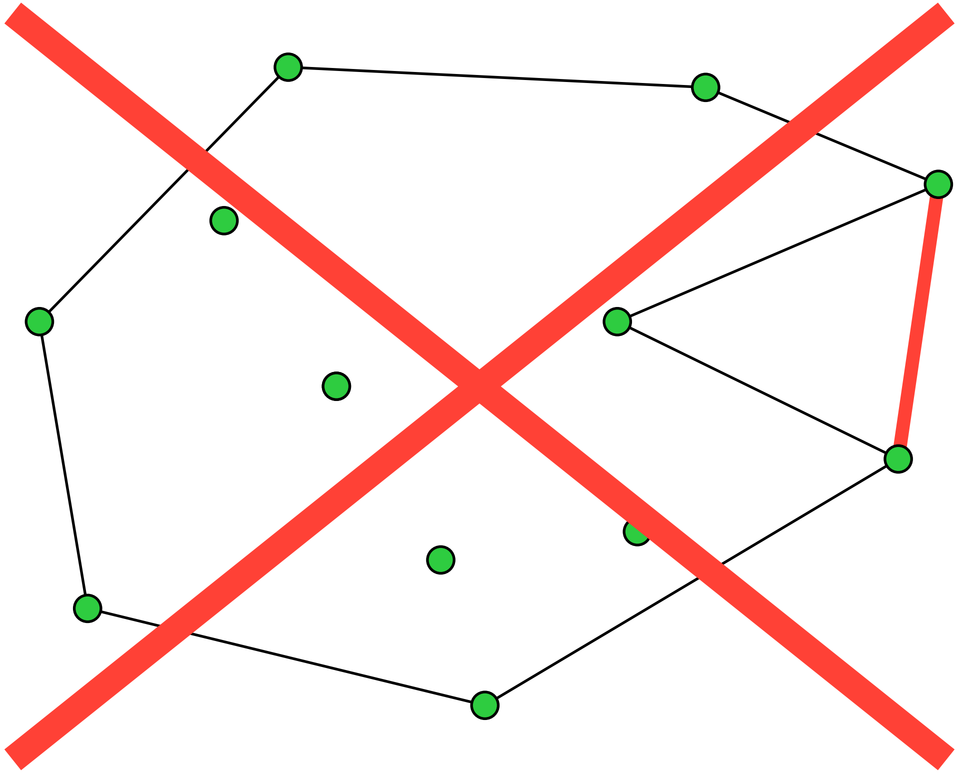
Otoczką wypukłą zbioru punktów nazywamy najmniejszy wielokąt wypukły zawierający ten zbiór. Będziemy oznaczać liczbę punktów zbioru wejściowego jako n, a liczbę punktów otoczki jako h.



1.1 Definicja

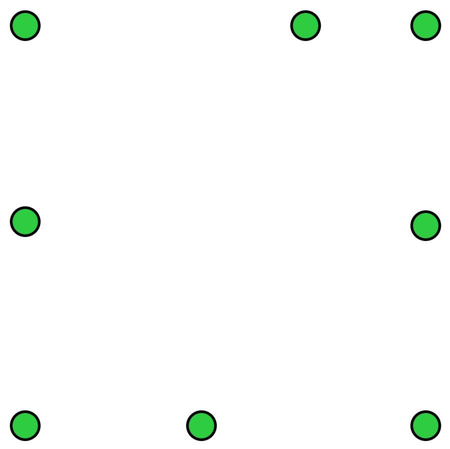
Otoczką wypukłą zbioru punktów nazywamy najmniejszy wielokąt wypukły zawierający ten zbiór. Będziemy oznaczać liczbę punktów zbioru wejściowego jako n, a liczbę punktów otoczki jako h.

Nie jest wypukła!



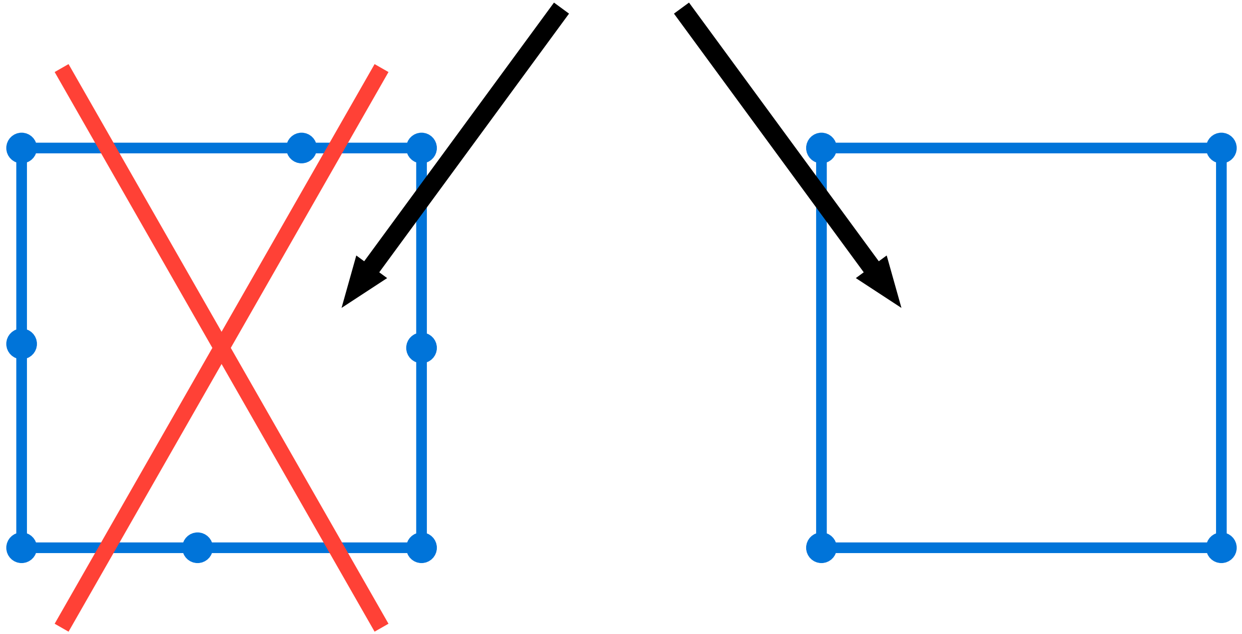
1.2 Poprawność

Otoczkę uznajemy za poprawną jeżeli jej wierzchołki są zadane w kierunku przeciwnym lub zgodnym do ruchu wskazówek zegara. W dodatku, do otoczki **nie zaliczamy wewnętrznych punktów współliniowych**.



1.2 Poprawność

Otoczkę uznajemy za poprawną jeżeli jej wierzchołki są zadane w kierunku przeciwnym lub zgodnym do ruchu wskazówek zegara. W dodatku, do otoczki **nie zaliczamy wewnętrznych punktów współliniowych**.



2. Algorytmy

2.1 Przyrostowy

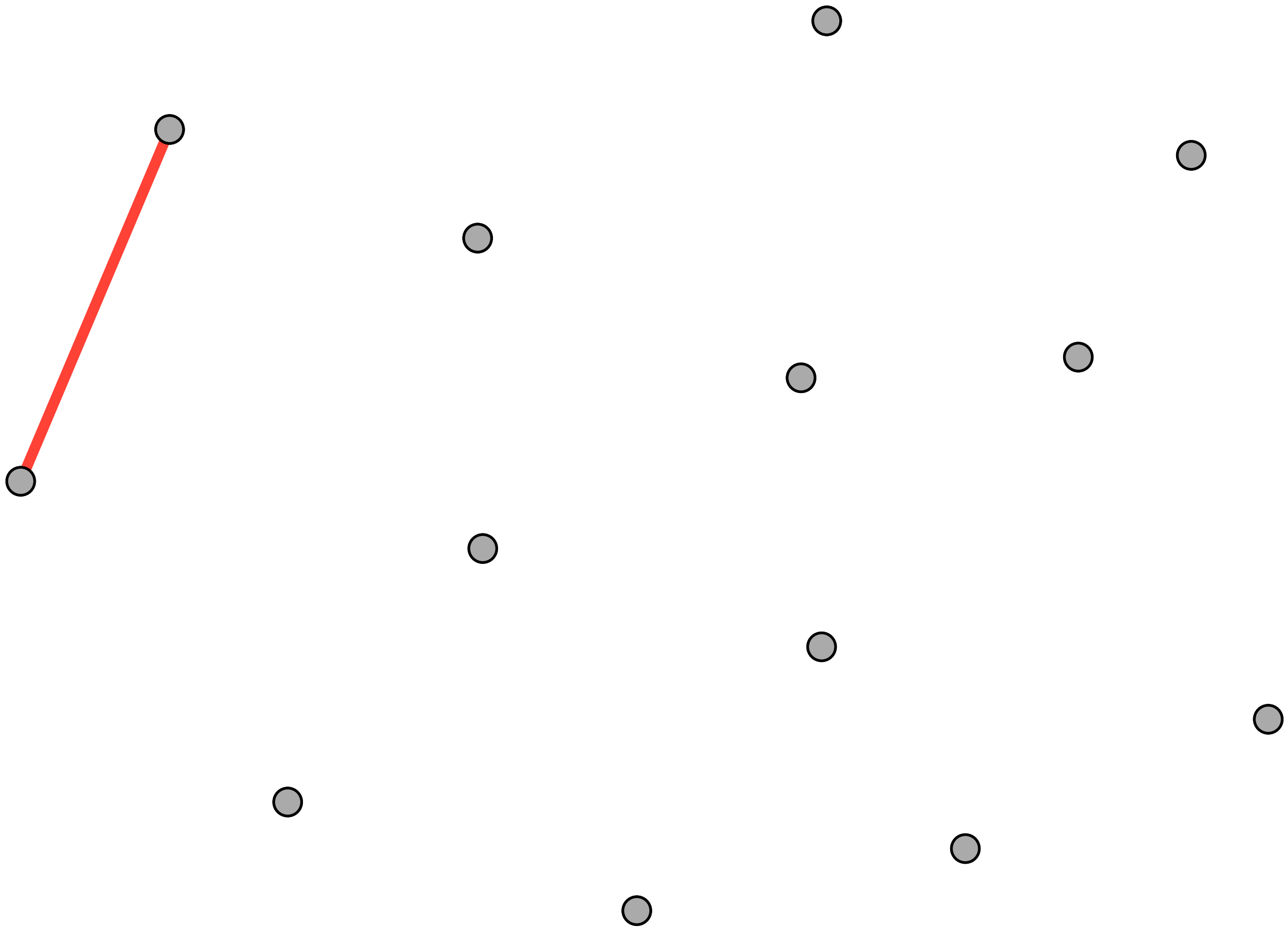
2.1.1 Działanie algorytmu

Algorytm najpierw sortuje punkty względem współrzędnej x, a następnie wybiera dwa pierwsze punkty, będące początkową otoczką. Następnie iteracyjnie dodaje kolejne punkty do otoczki, dokonane jest to poprzez znalezienie **stycznych do otoczki przechodzących przez kolejne punkty** i odpowiednie usuwanie punktów wewnątrz niej.

Złożoność czasowa algorytmu to $O(n \log n)$

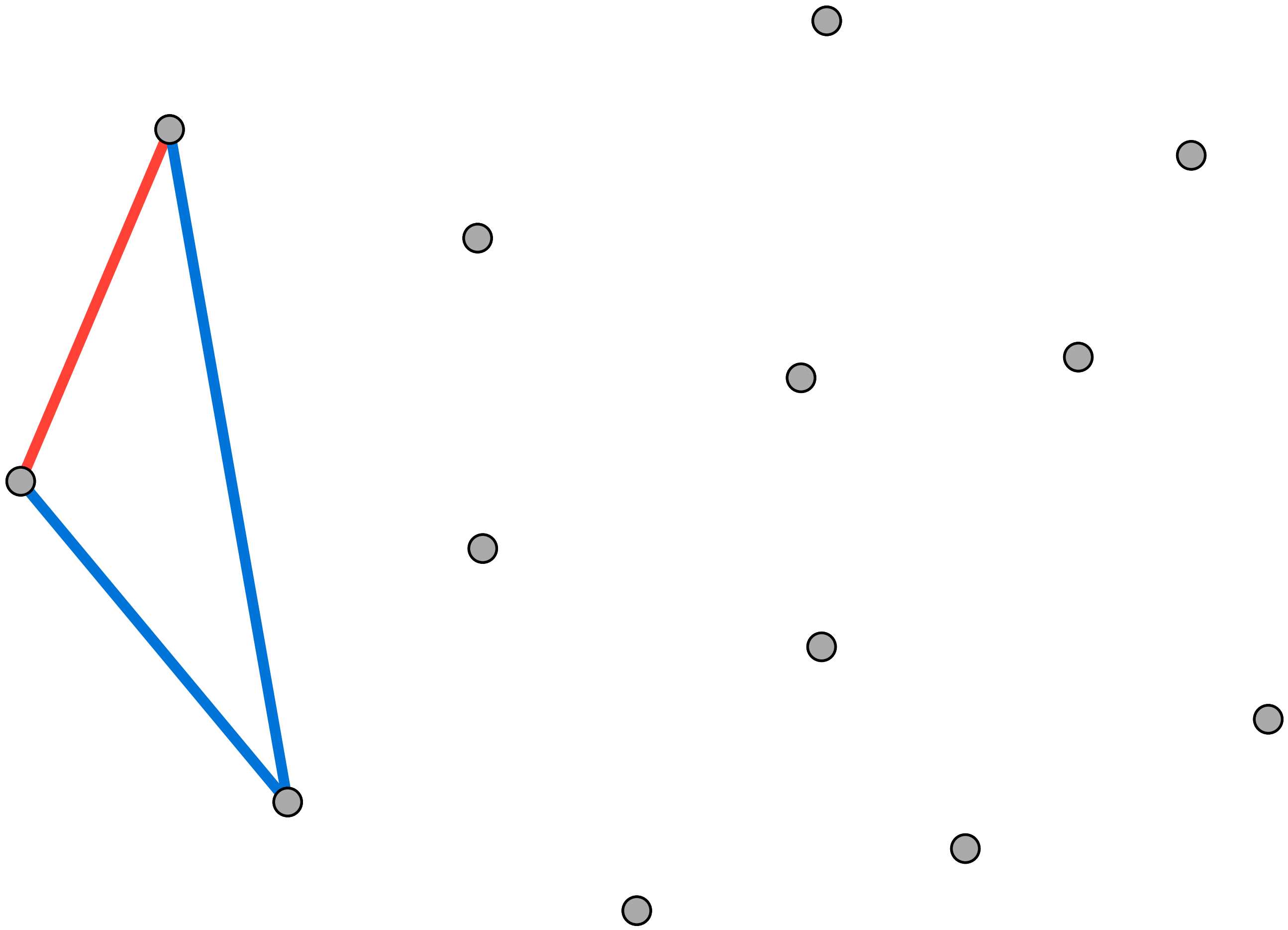
2.1 Przyrostowy

2.1.2 Przykład



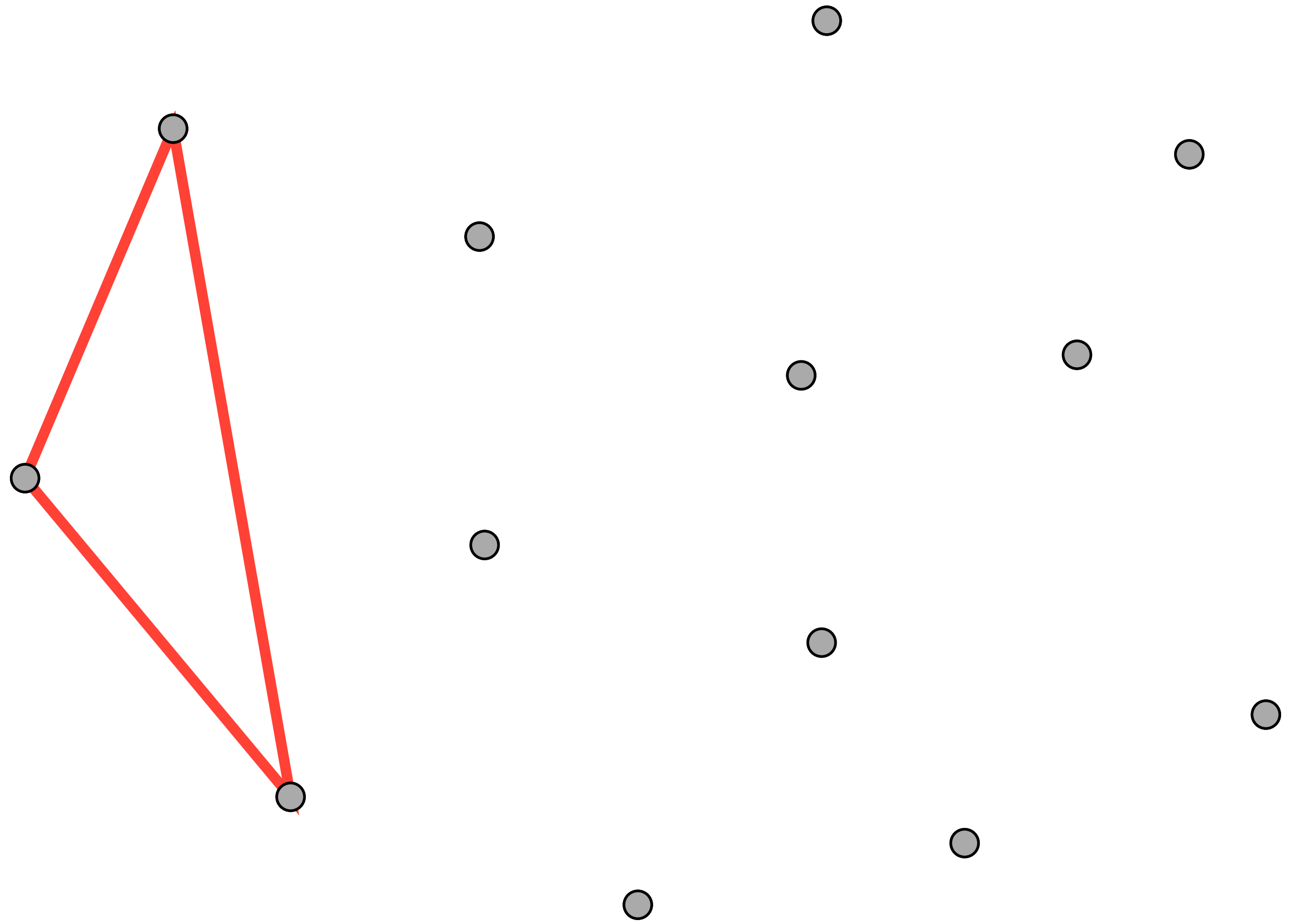
2.1 Przyrostowy

2.1.2 Przykład



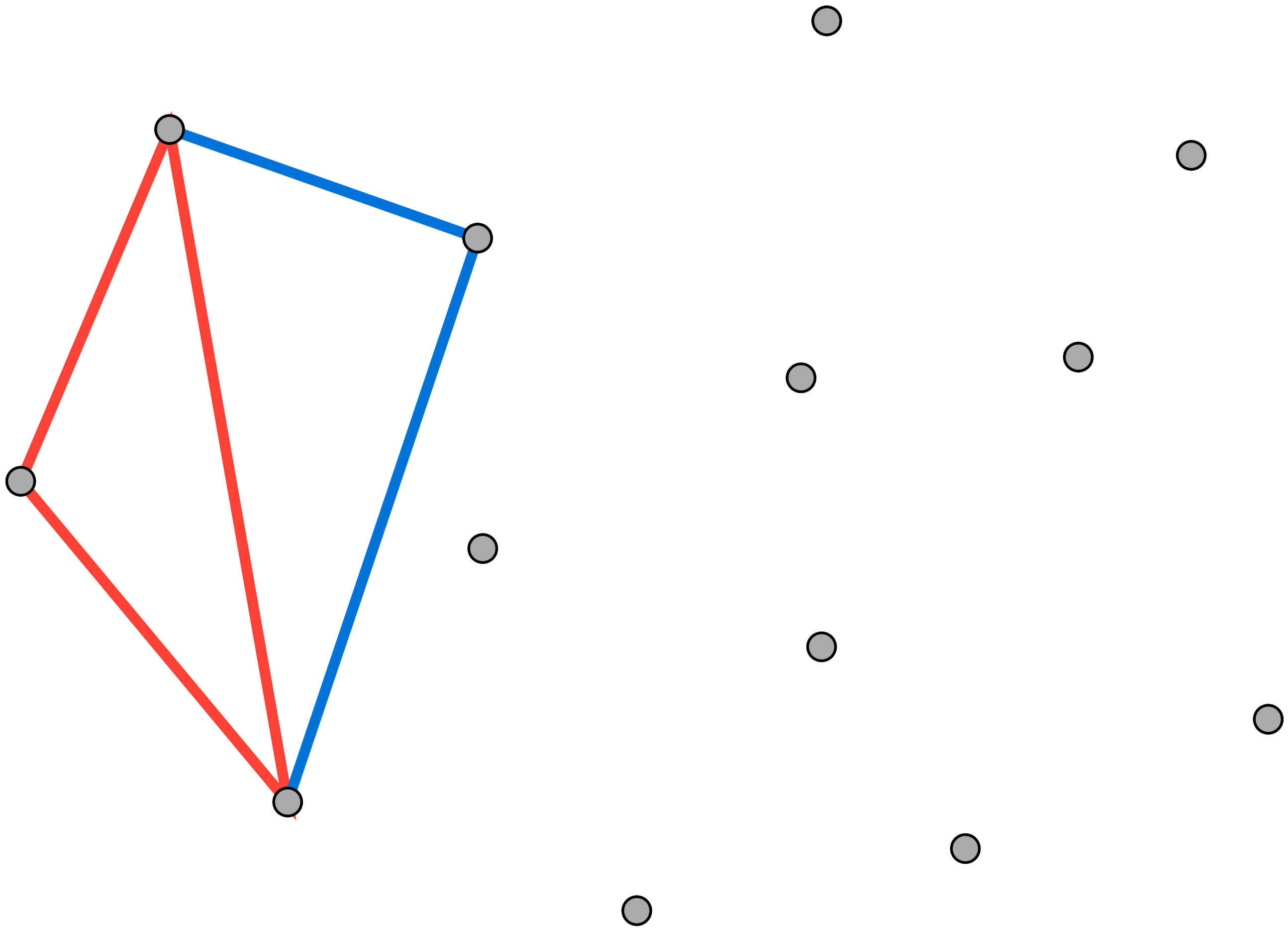
2.1 Przyrostowy

2.1.2 Przykład



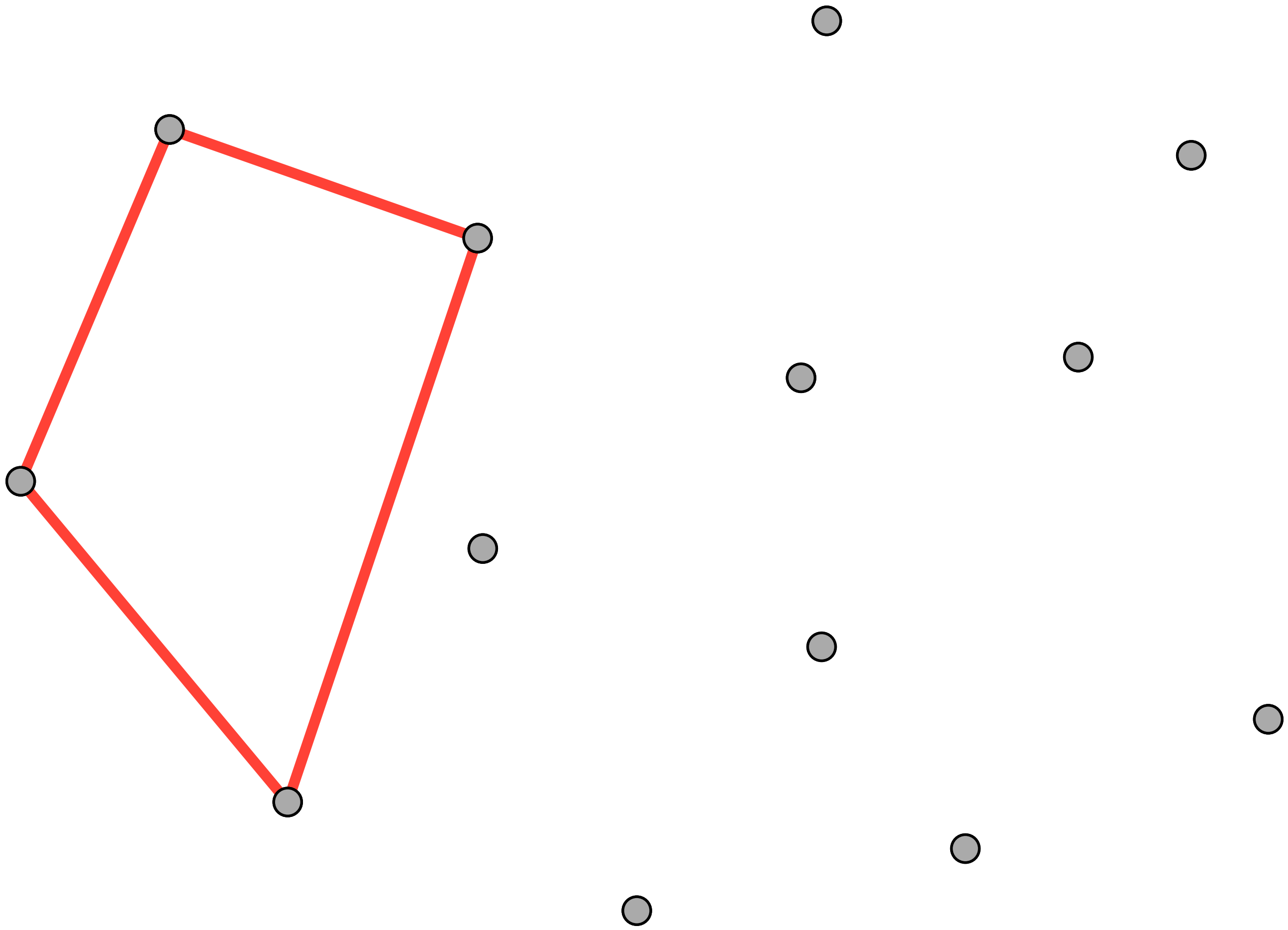
2.1 Przyrostowy

2.1.2 Przykład



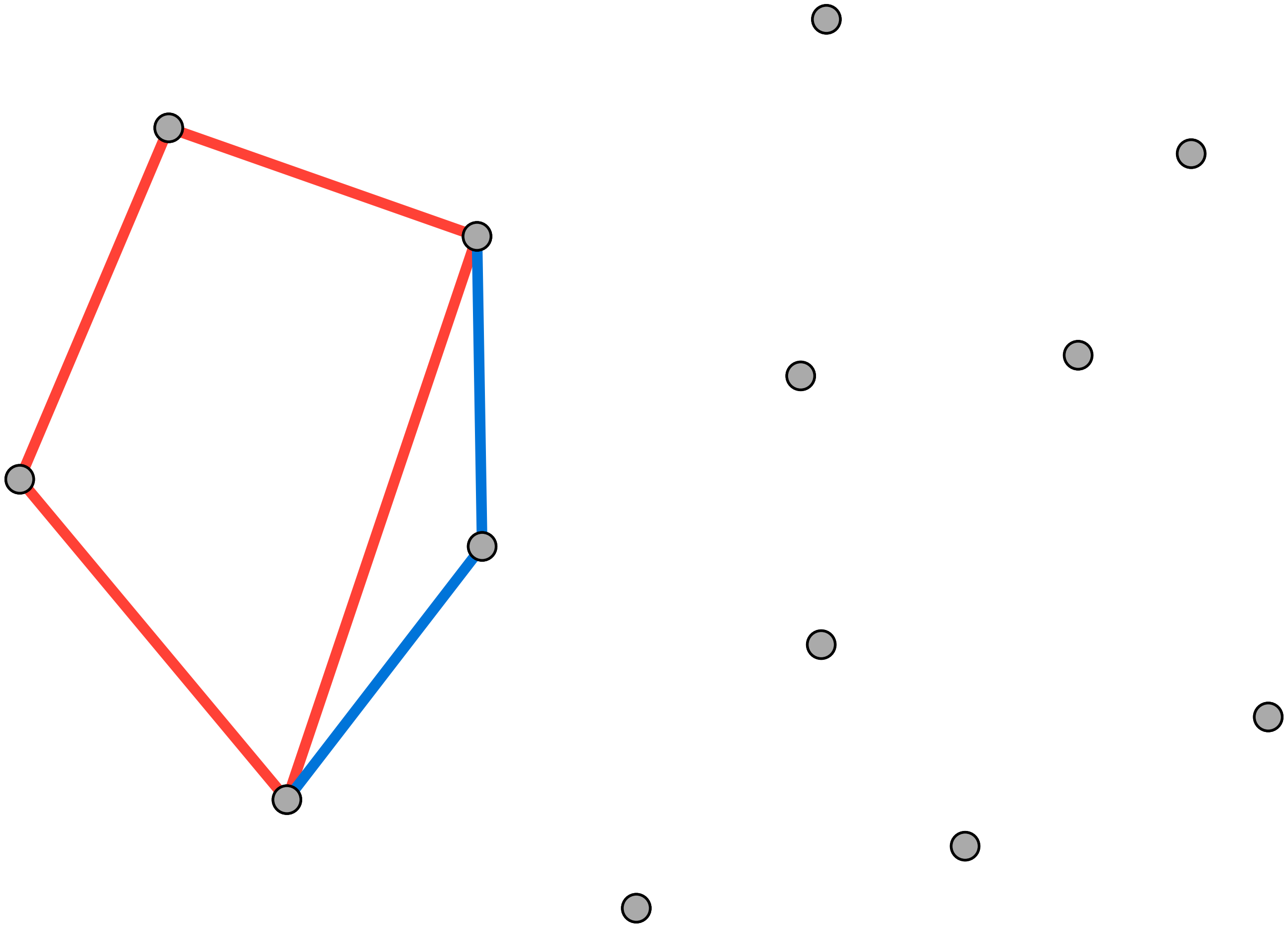
2.1 Przyrostowy

2.1.2 Przykład



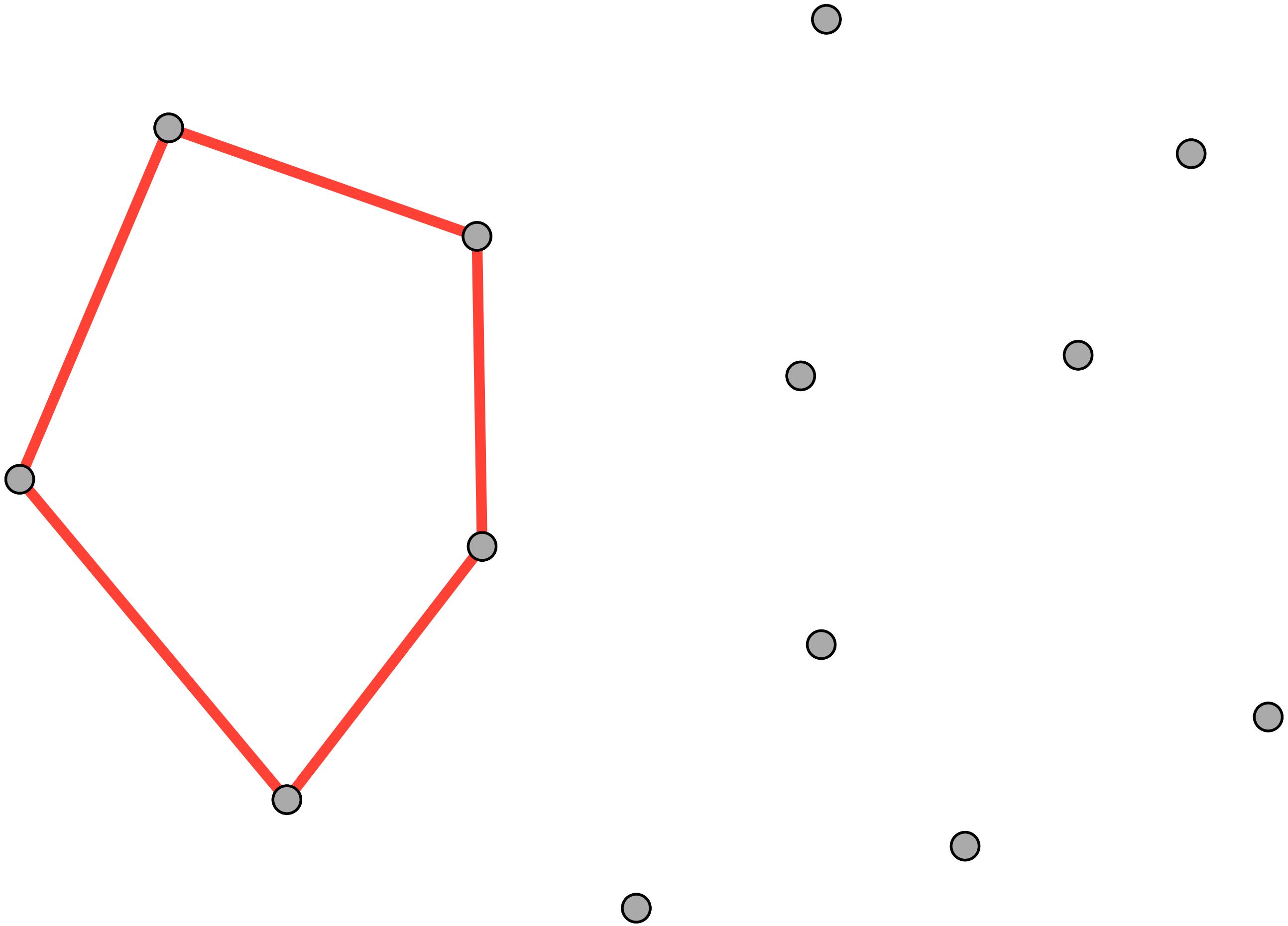
2.1 Przyrostowy

2.1.2 Przykład



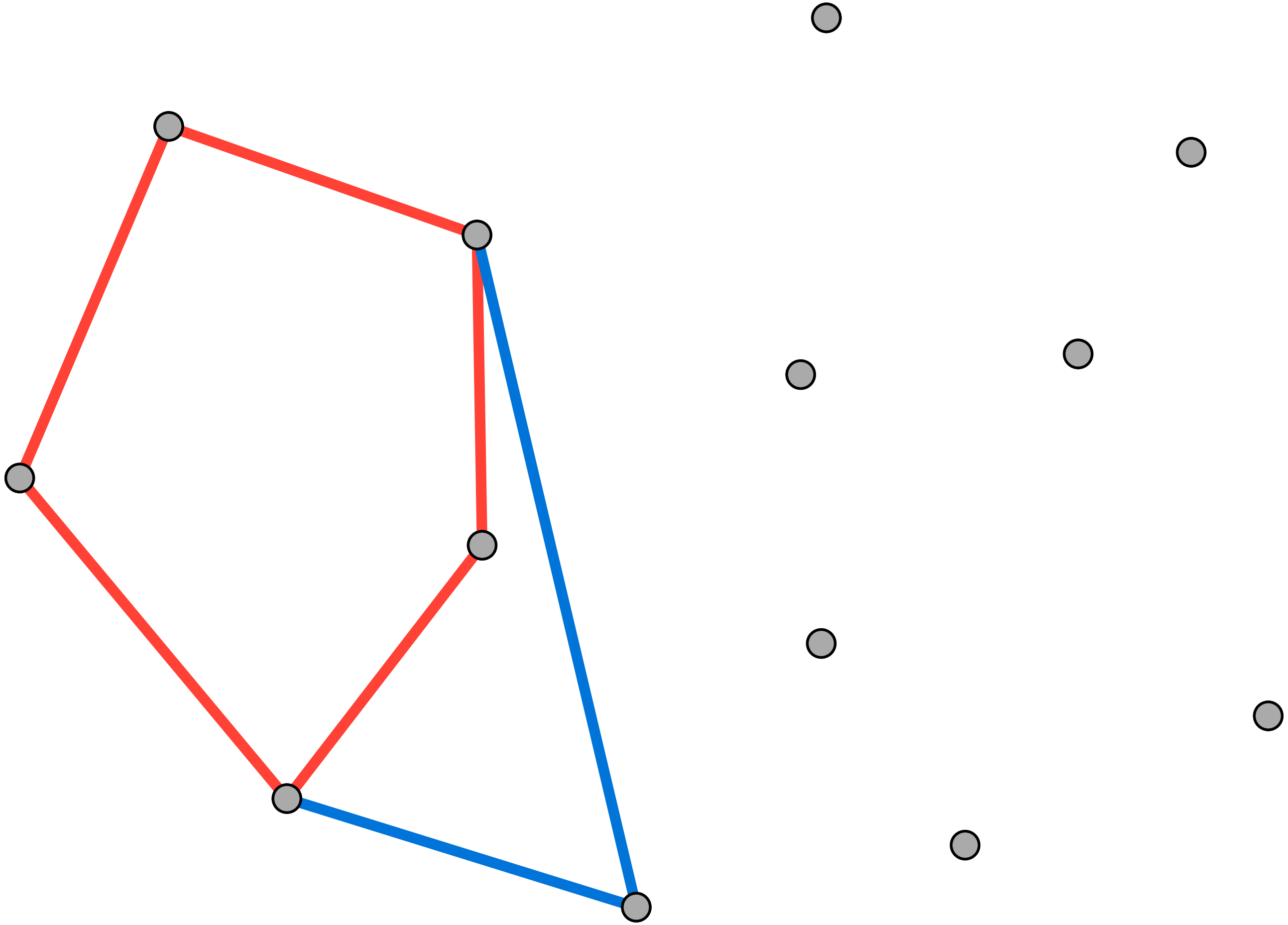
2.1 Przyrostowy

2.1.2 Przykład



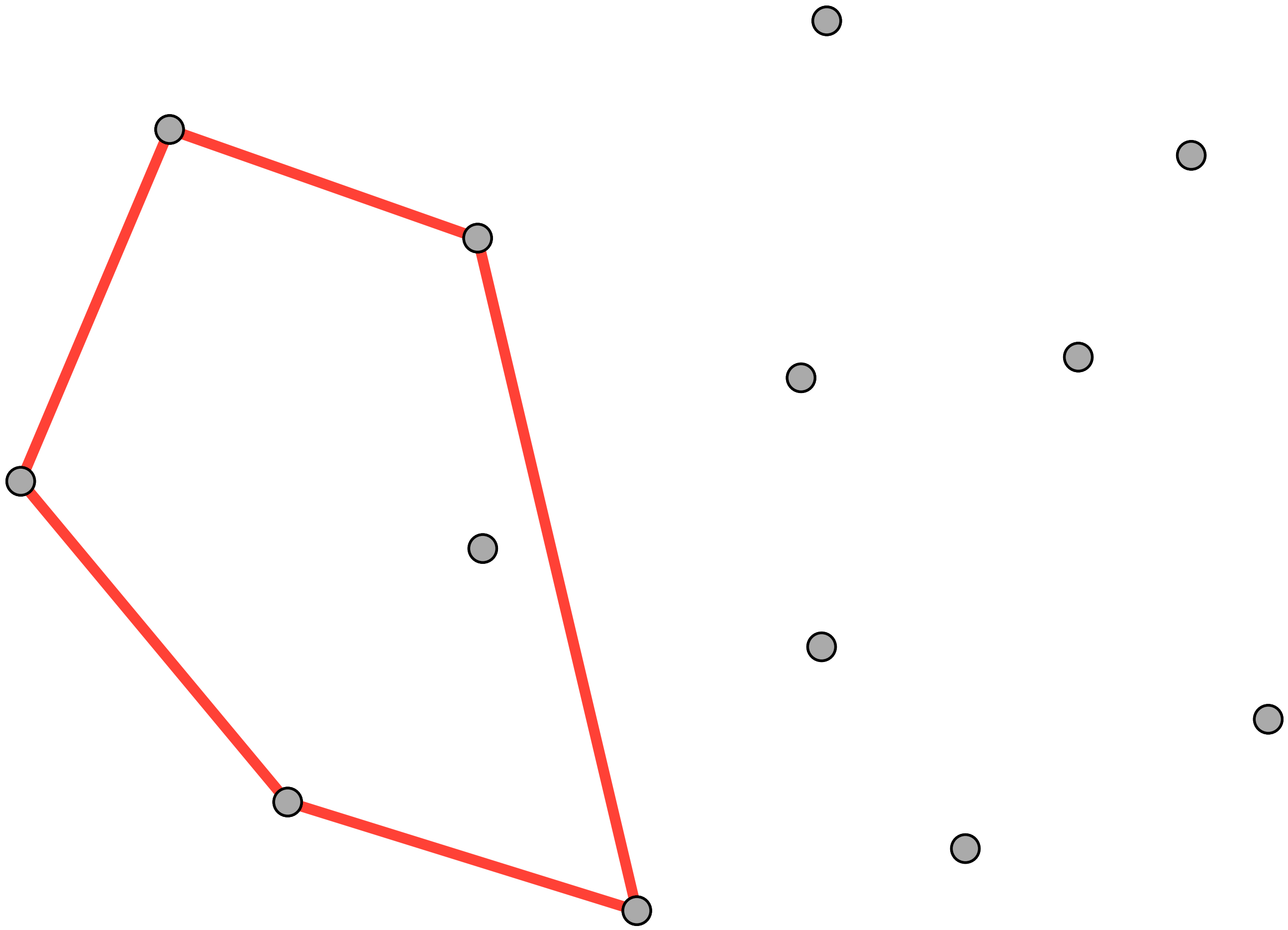
2.1 Przyrostowy

2.1.2 Przykład



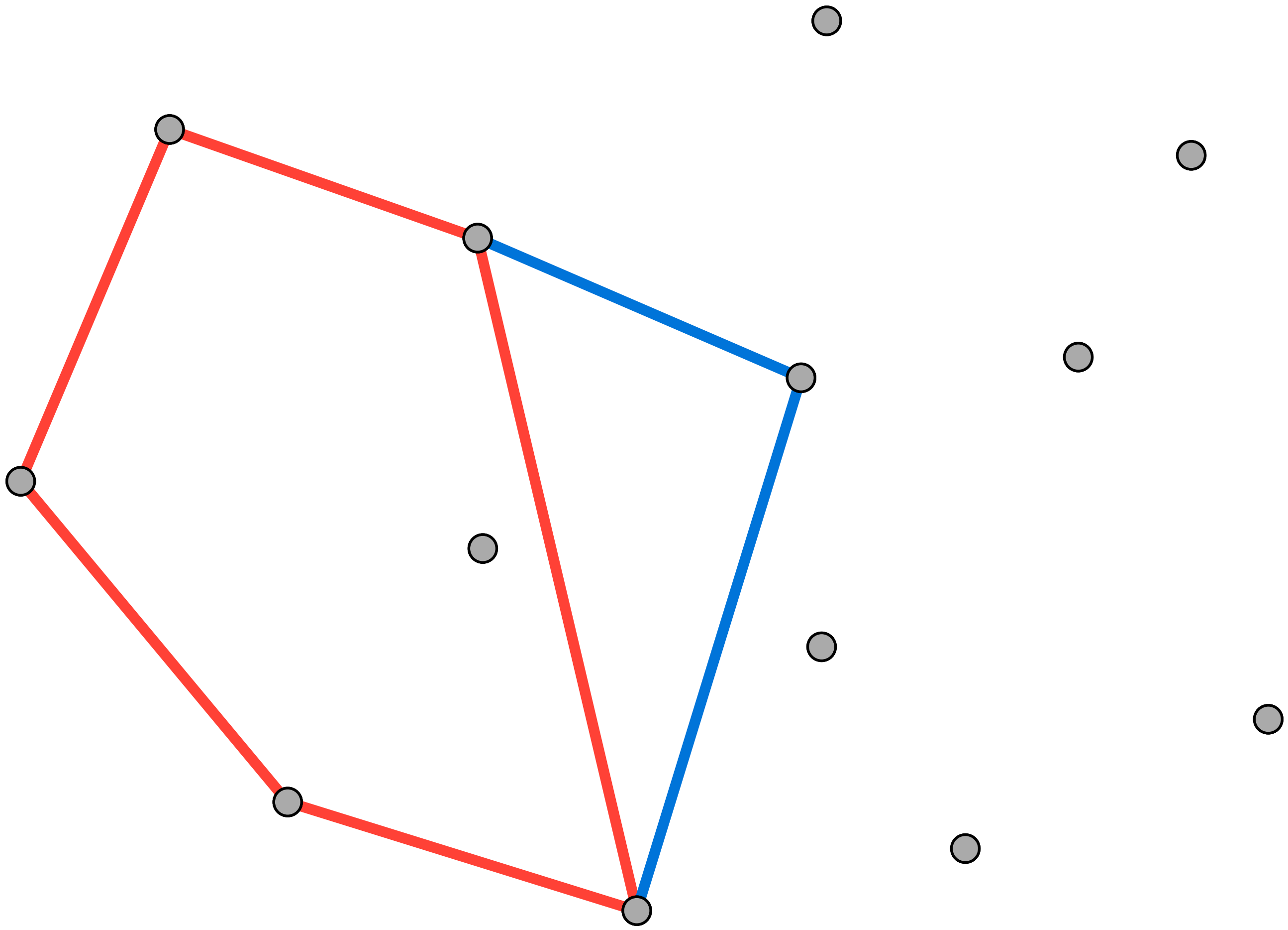
2.1 Przyrostowy

2.1.2 Przykład



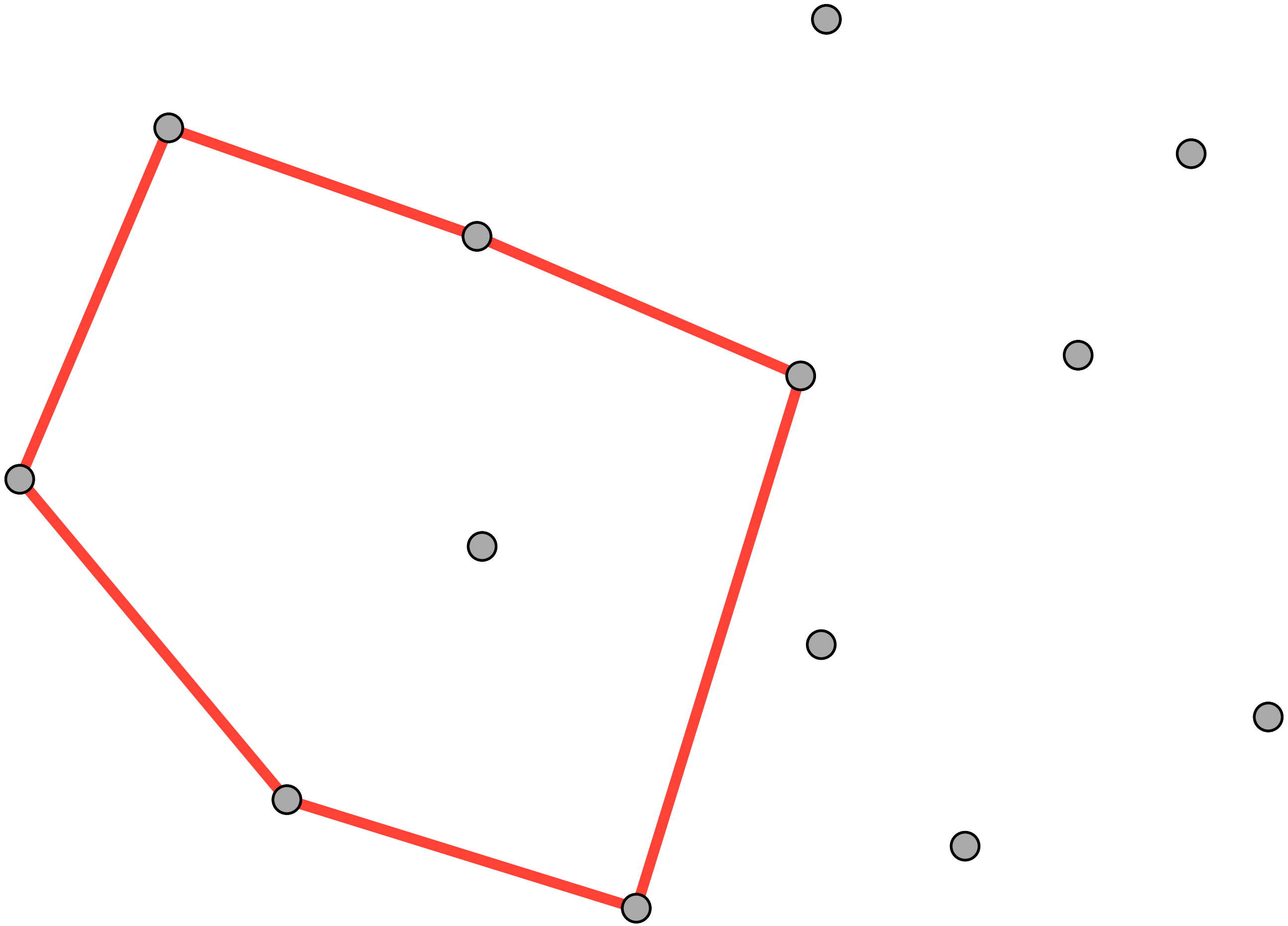
2.1 Przyrostowy

2.1.2 Przykład



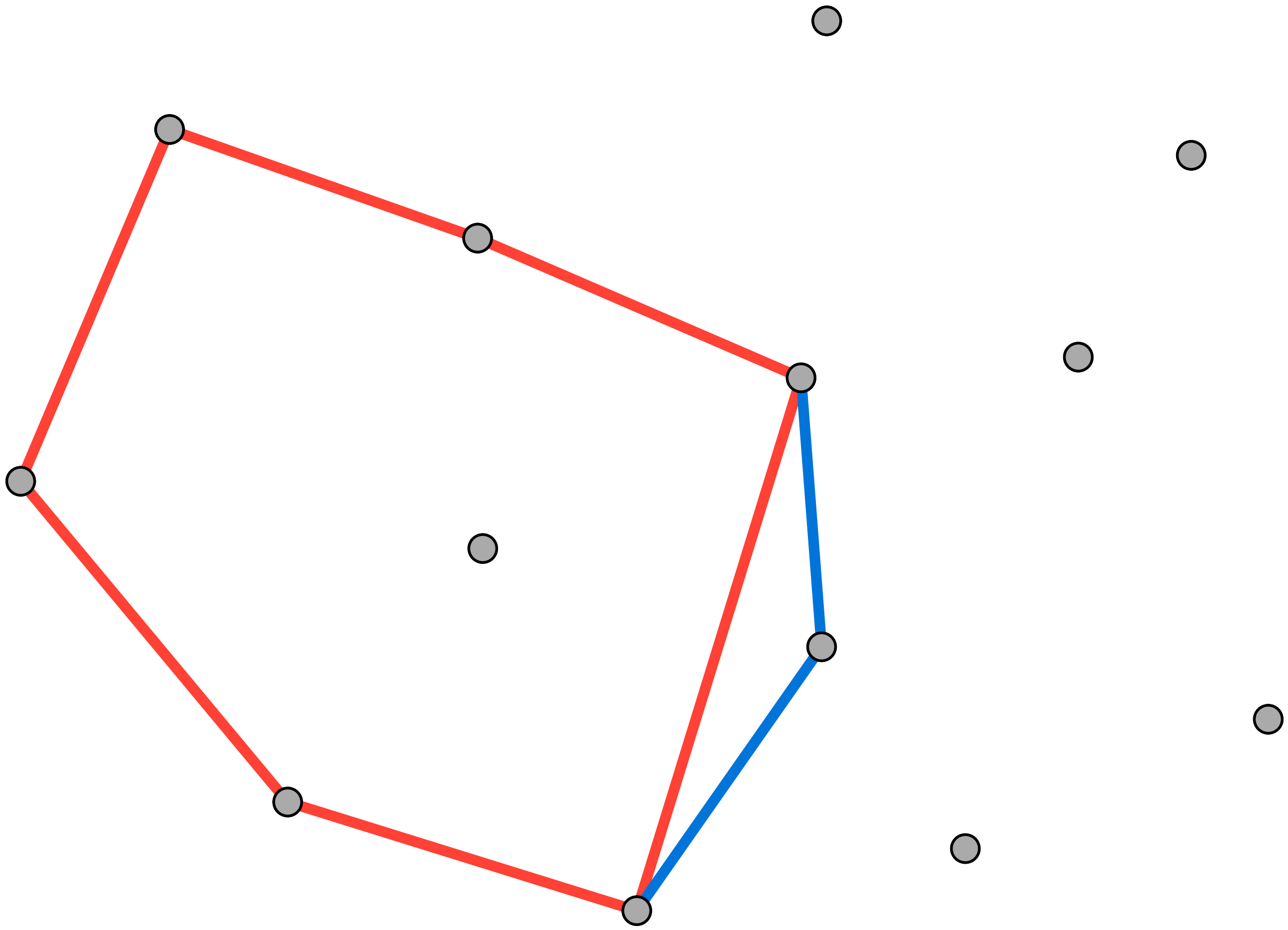
2.1 Przyrostowy

2.1.2 Przykład



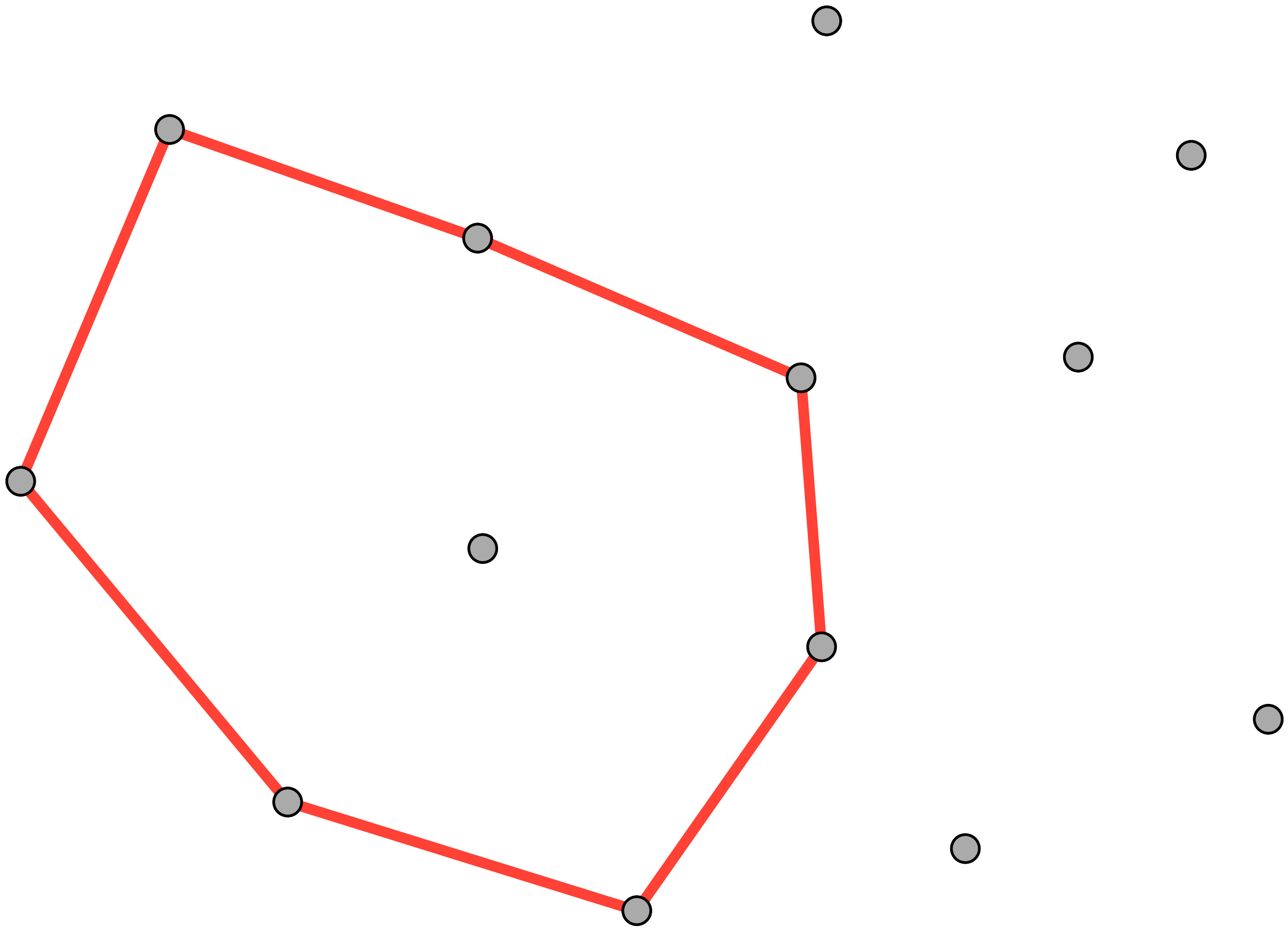
2.1 Przyrostowy

2.1.2 Przykład



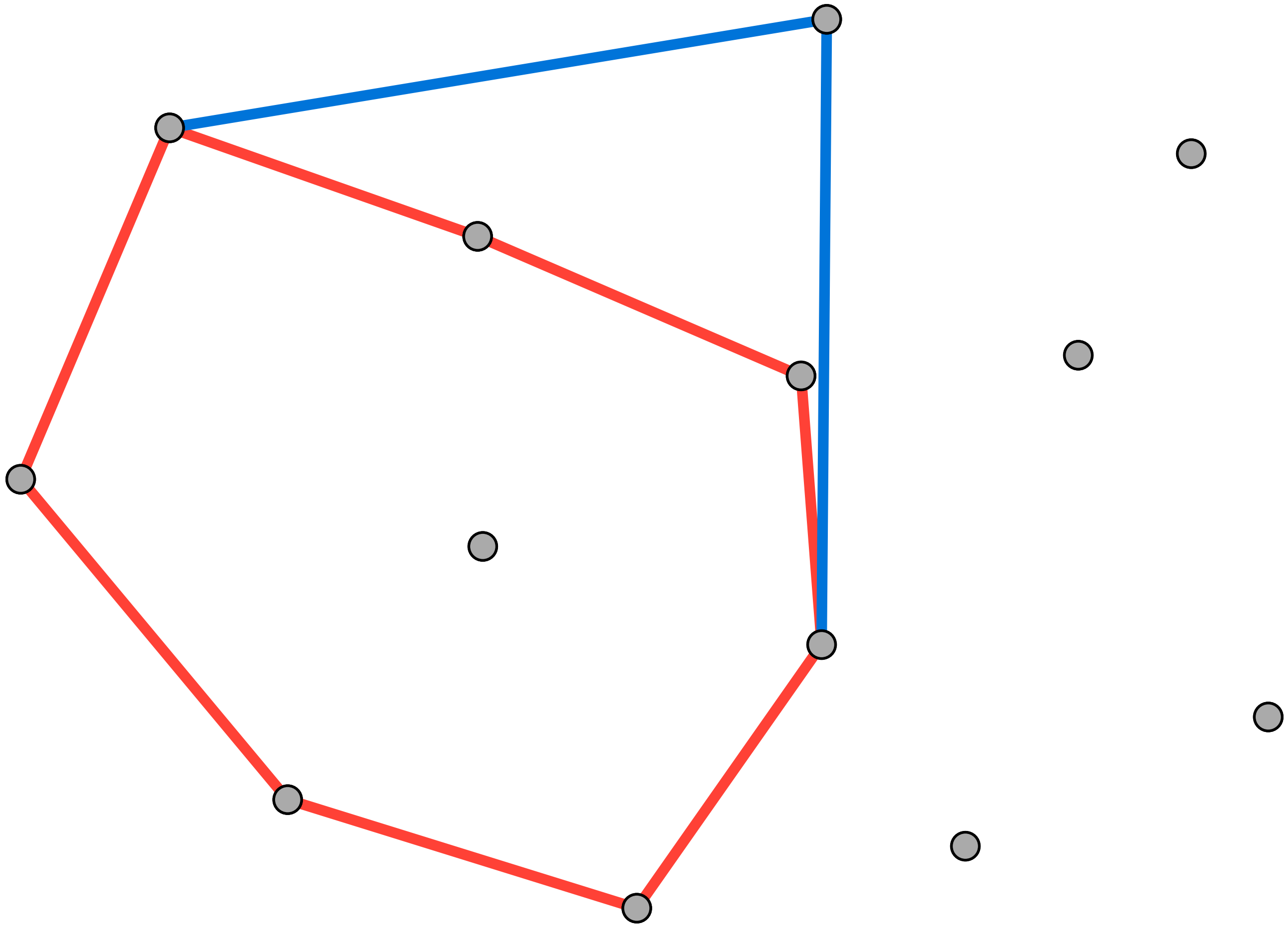
2.1 Przyrostowy

2.1.2 Przykład



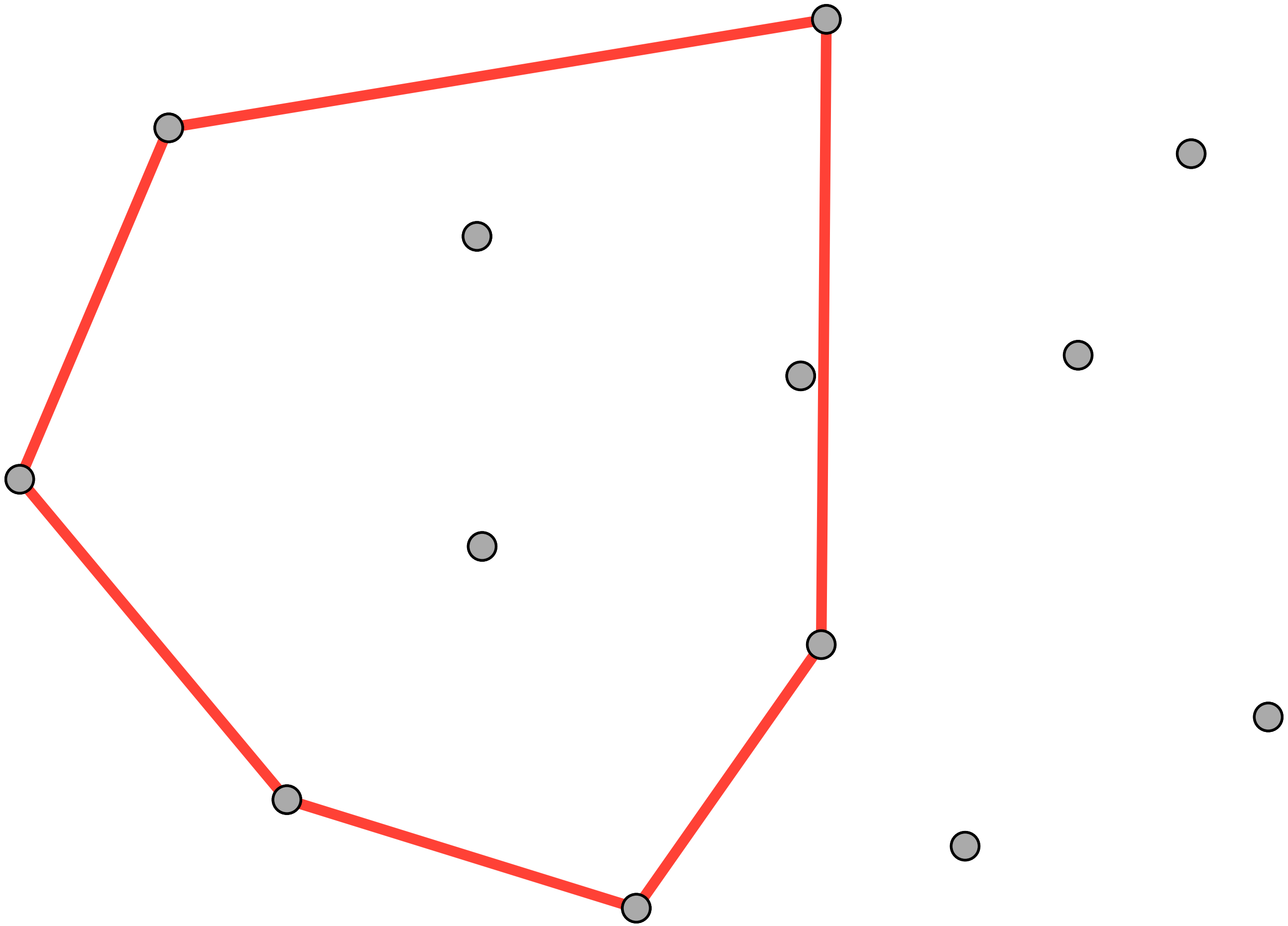
2.1 Przyrostowy

2.1.2 Przykład



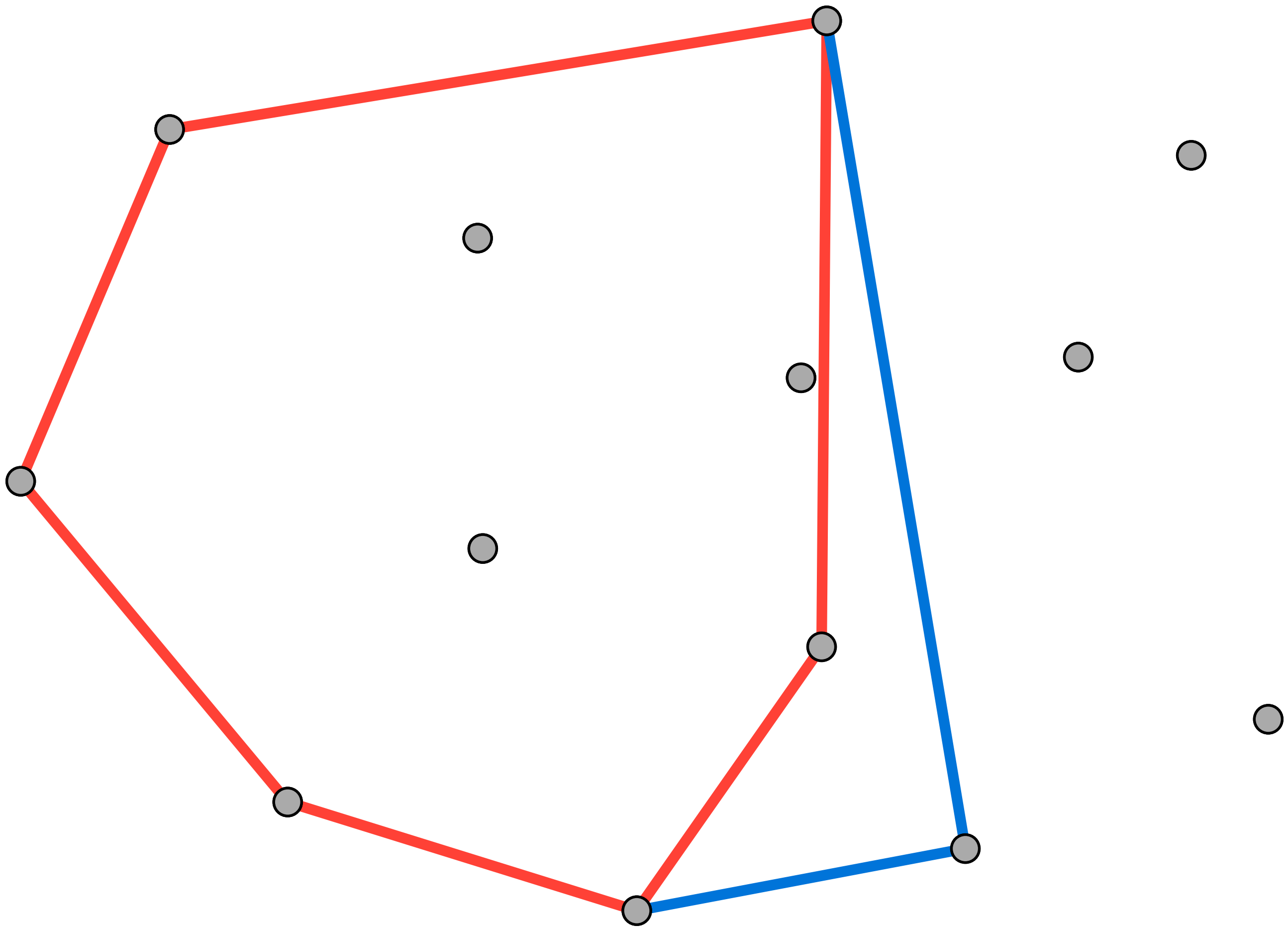
2.1 Przyrostowy

2.1.2 Przykład



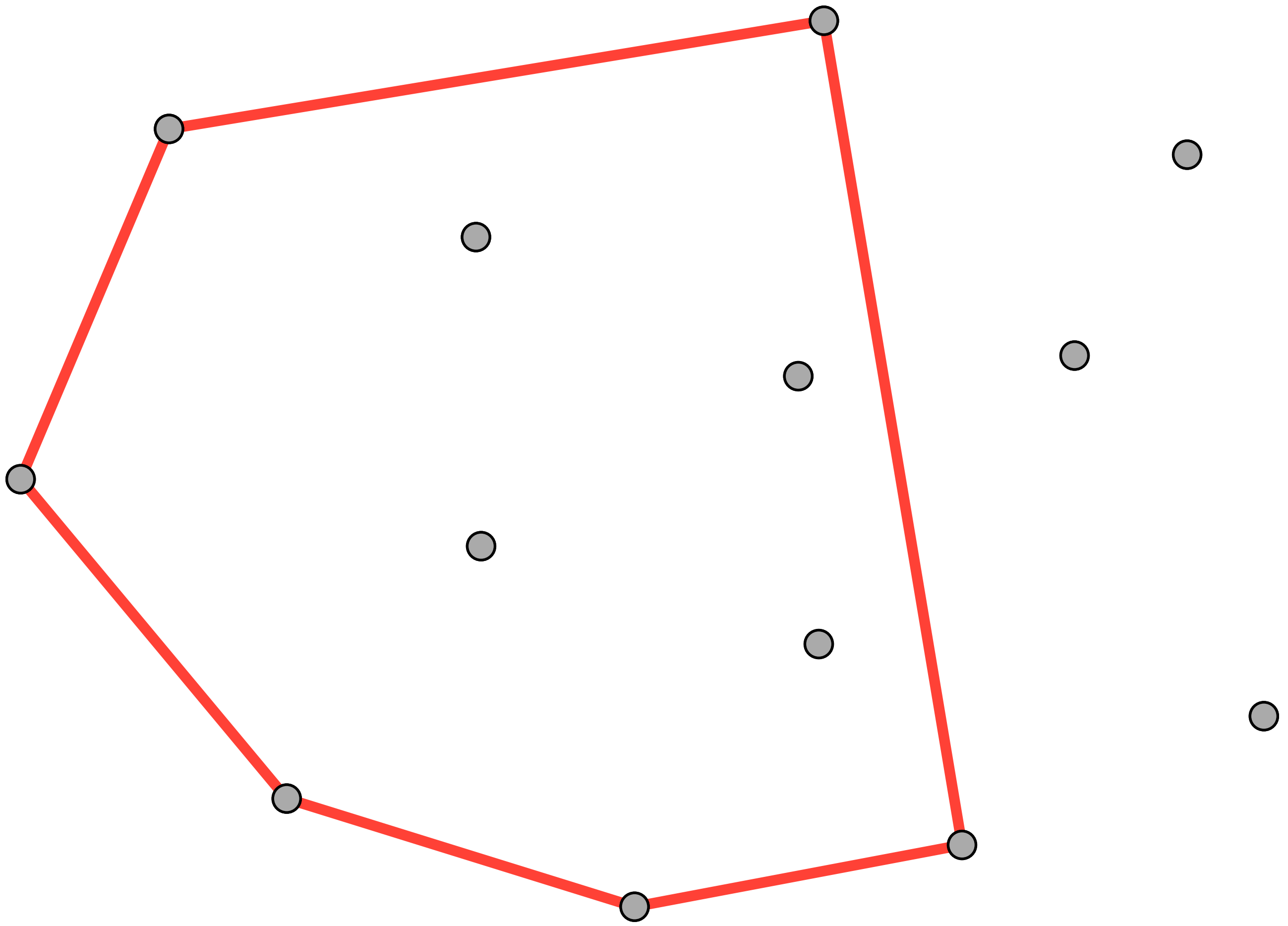
2.1 Przyrostowy

2.1.2 Przykład



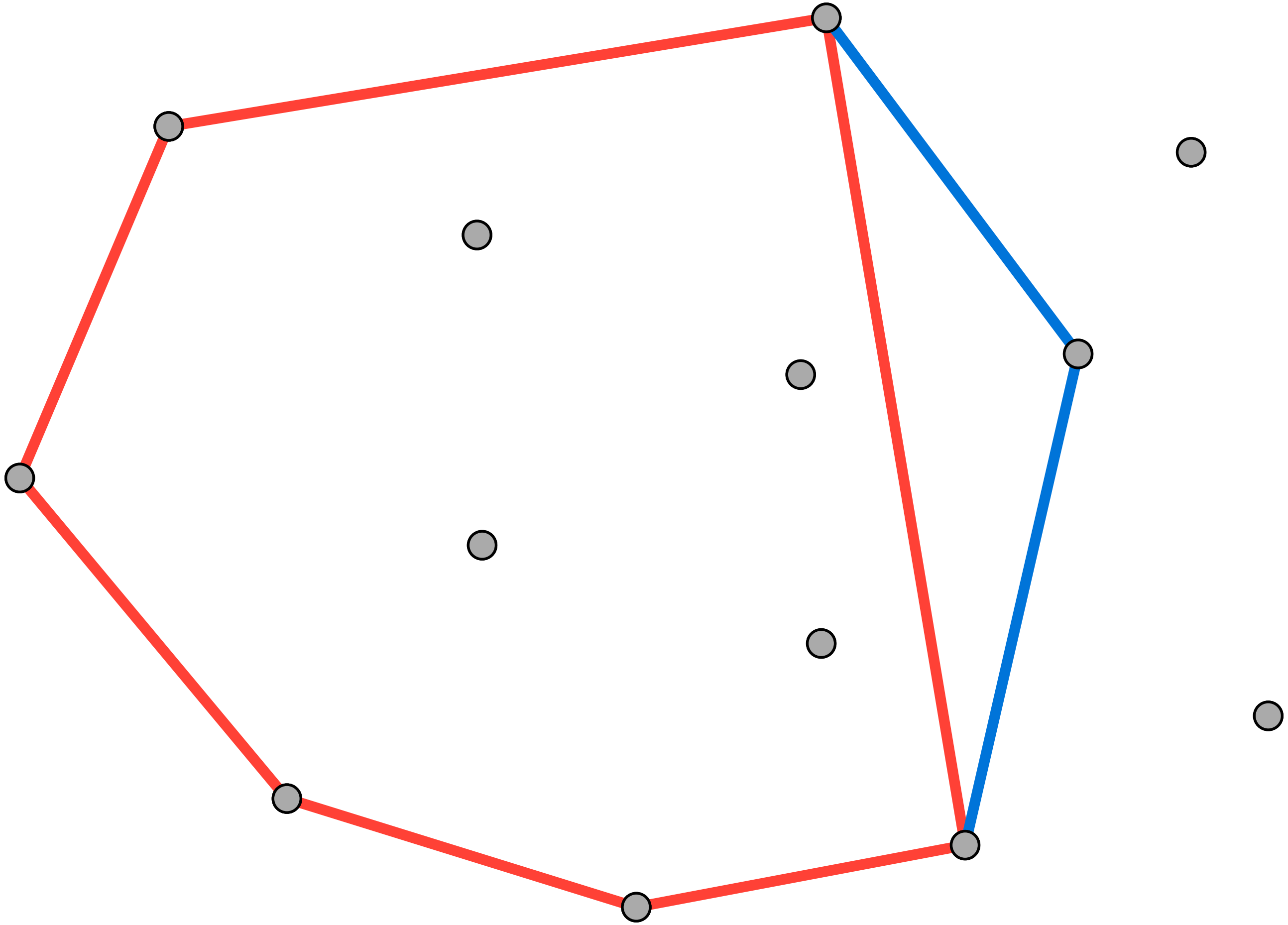
2.1 Przyrostowy

2.1.2 Przykład



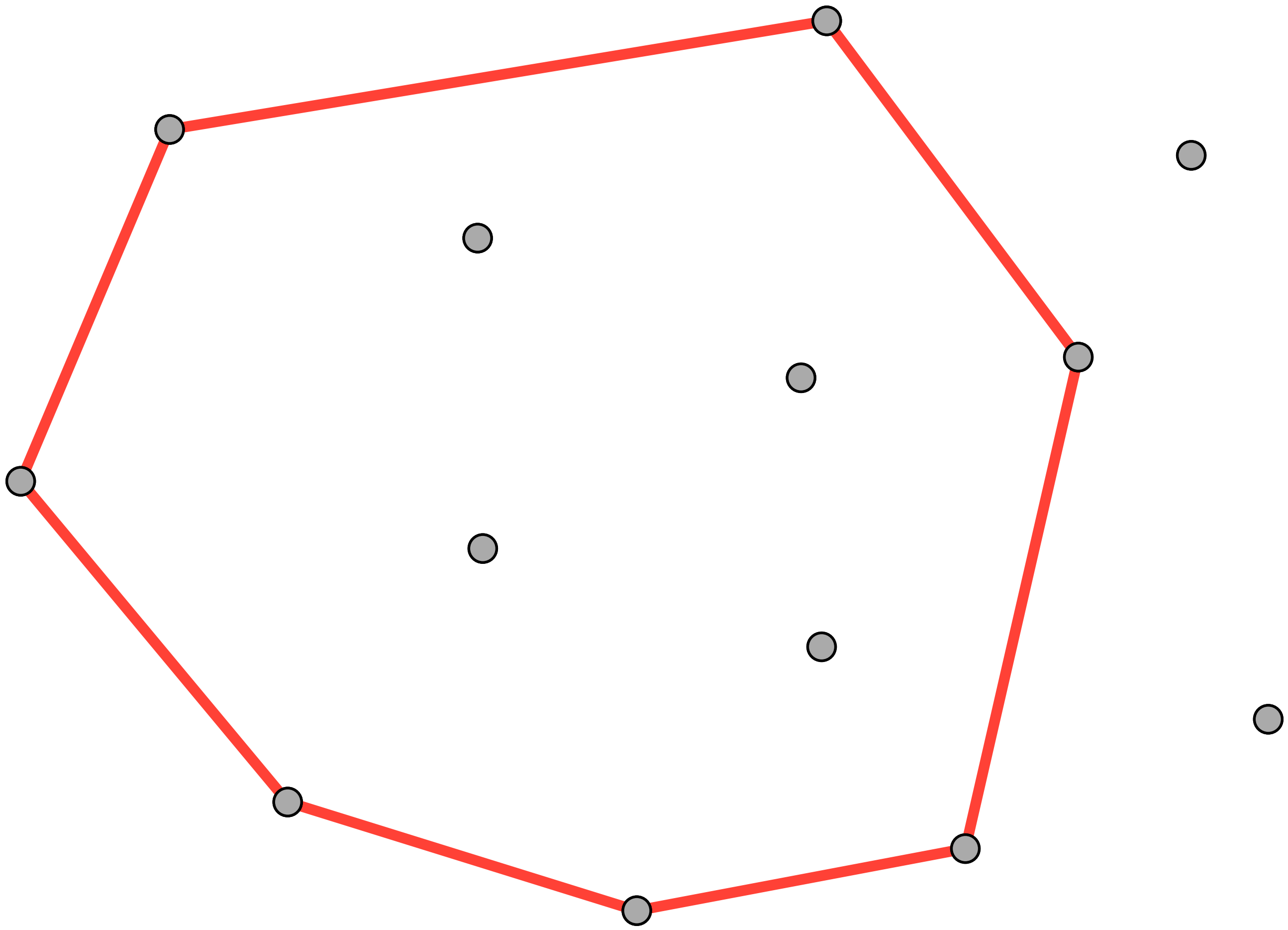
2.1 Przyrostowy

2.1.2 Przykład



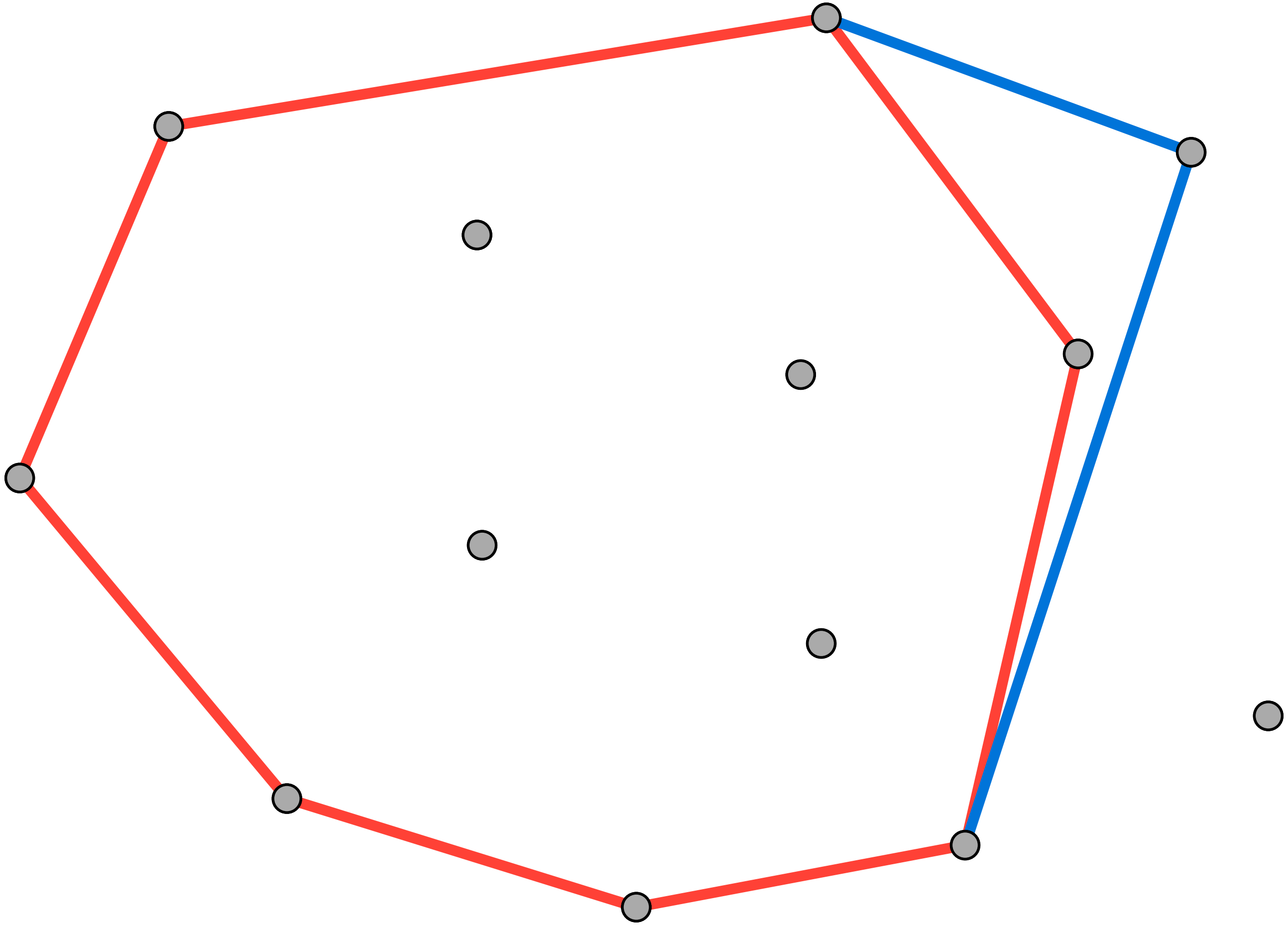
2.1 Przyrostowy

2.1.2 Przykład



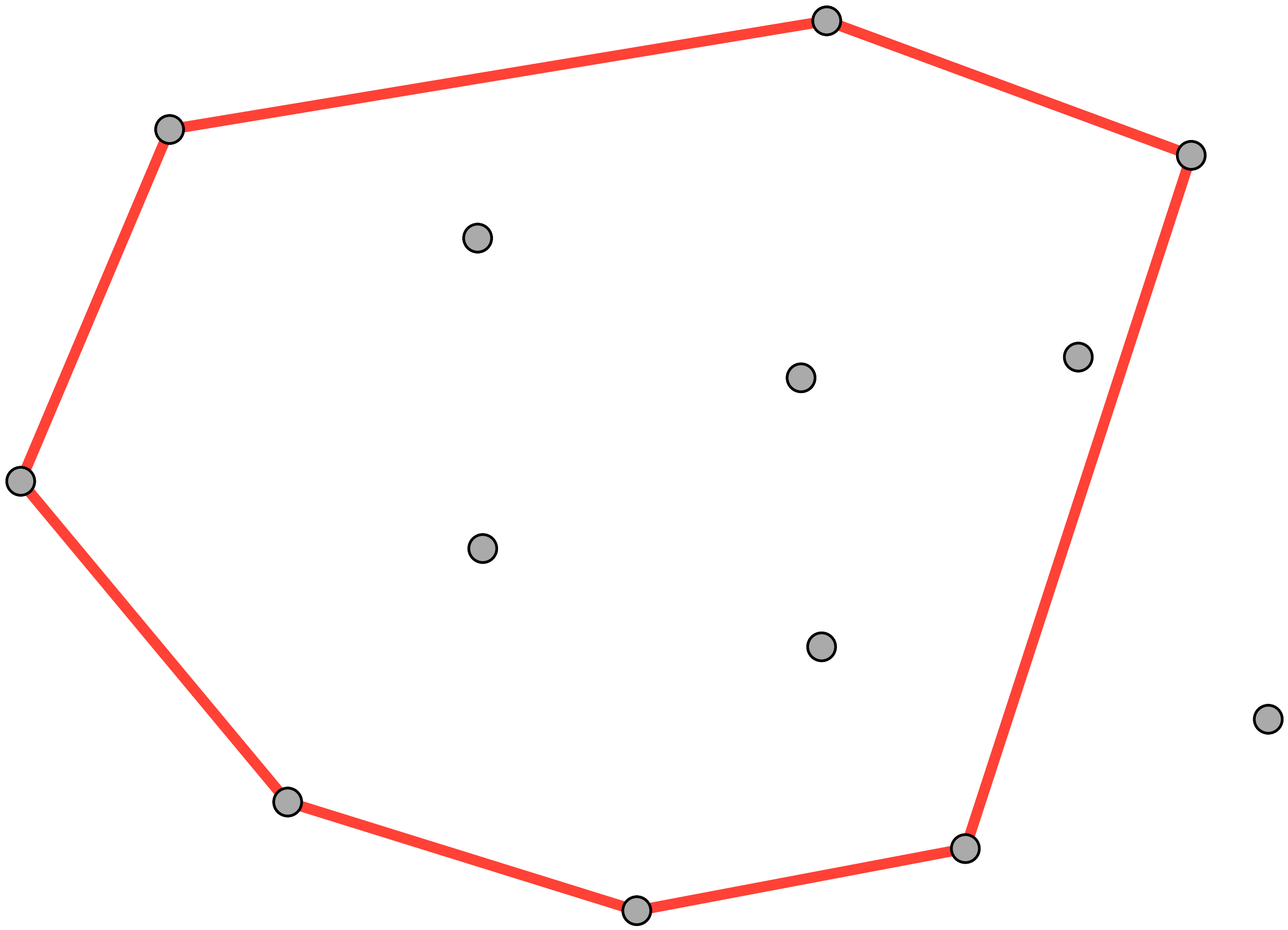
2.1 Przyrostowy

2.1.2 Przykład



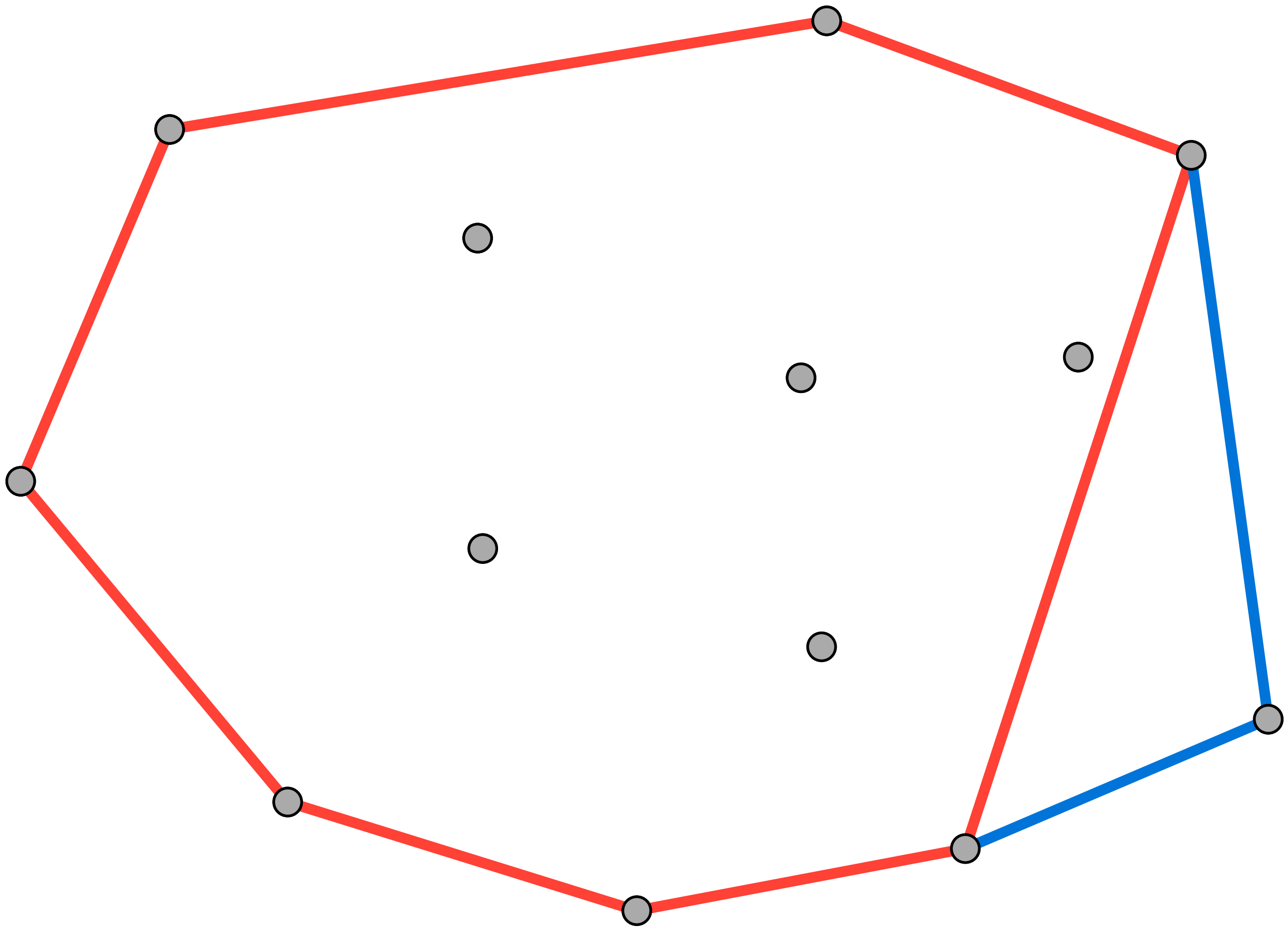
2.1 Przyrostowy

2.1.2 Przykład



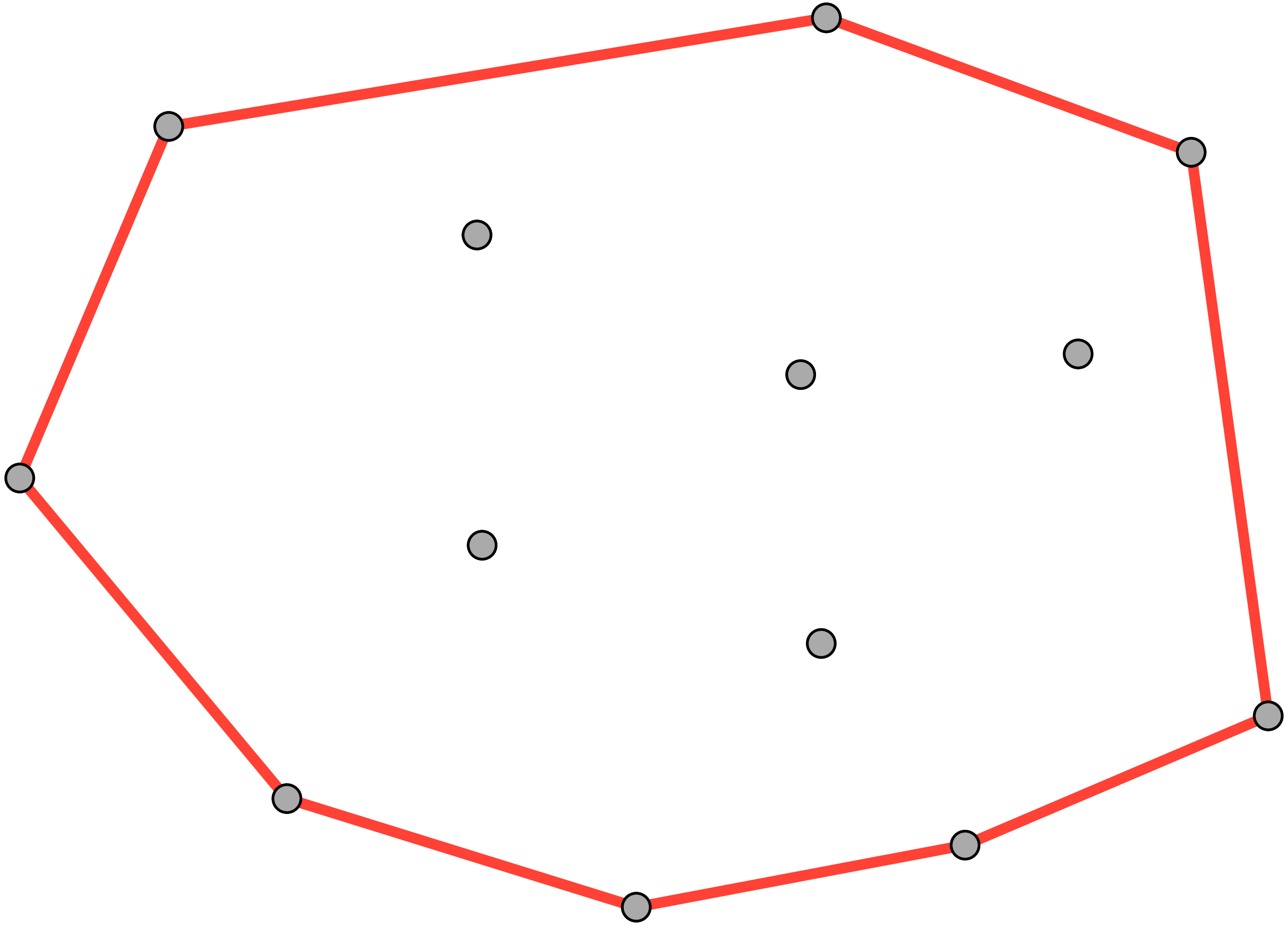
2.1 Przyrostowy

2.1.2 Przykład



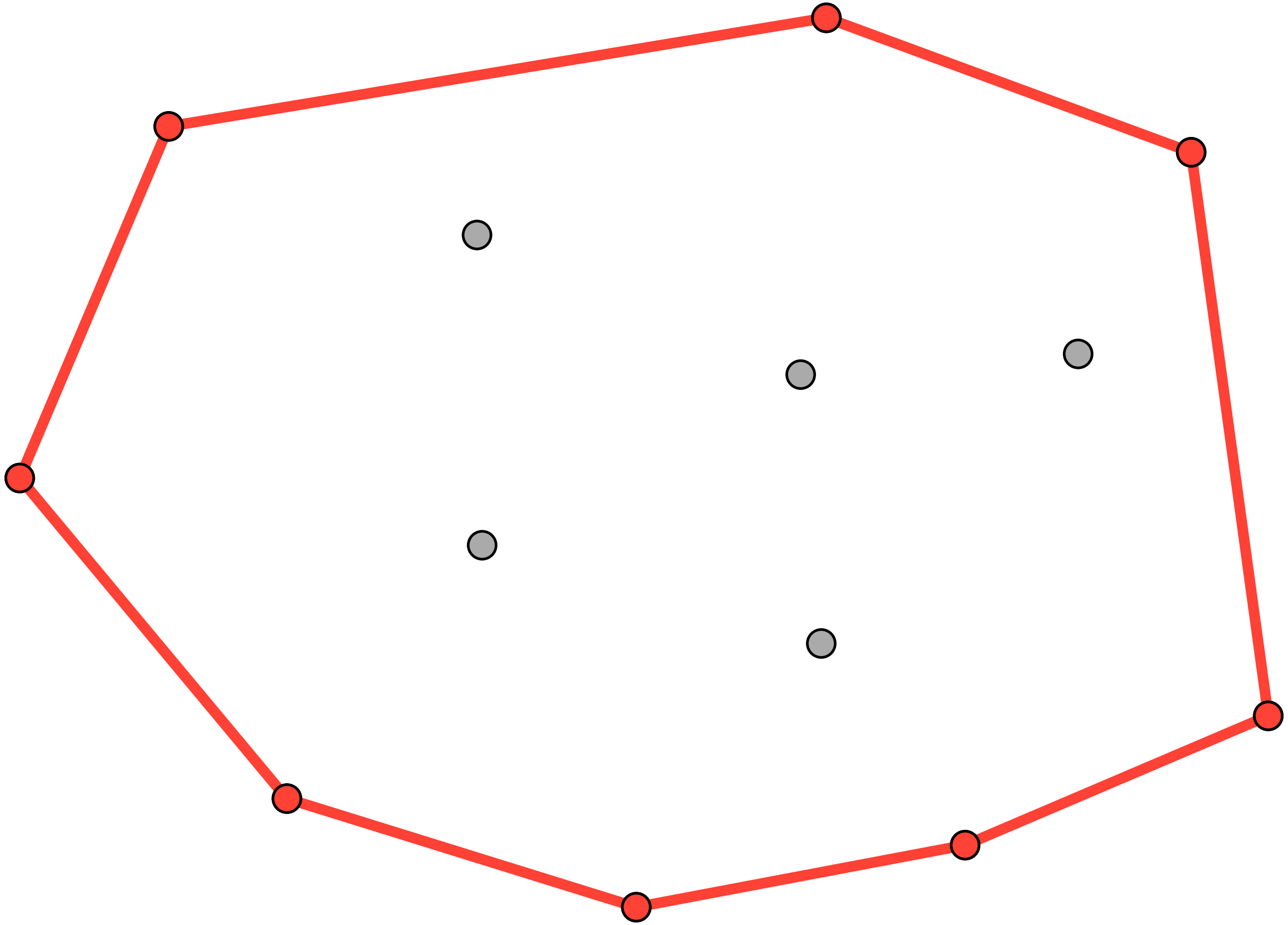
2.1 Przyrostowy

2.1.2 Przykład



2.1 Przyrostowy

2.1.2 Przykład



2.8 Simple Animation

We can use #pause to

Meanwhile,

2.8 Simple Animation

We can use `#pause` to display something later.

Meanwhile, we can also use `#meanwhile` to

2.8 Simple Animation

We can use `#pause` to display something later.

Just like this.

Meanwhile, we can also use `#meanwhile` to display other content synchronously.

2.9 Complex Animation

At subslide 1, we can

use for reserving space,

use for not reserving space,

call `#only` multiple times ✗ for choosing one of the alternatives.

2.9 Complex Animation

At subslide 2, we can

use `#uncover` function for reserving space,

use `#only` function for not reserving space,

use `#alternatives` function ✓ for choosing one of the alternatives.

2.10 Callback Style Animation

At subslide 1, we can

use for reserving space,

use for not reserving space,

call `#only` multiple times ✗ for choosing one of the alternatives.

2.10 Callback Style Animation

At subslide 2, we can

use `#uncover` function for reserving space,

use `#only` function for not reserving space,

use `#alternatives` function ✓ for choosing one of the alternatives.

2.10 Callback Style Animation

At subslide 3, we can

use `#uncover` function for reserving space,

use `#only` function for not reserving space,

use `#alternatives` function ✓ for choosing one of the alternatives.

2.11 Math Equation Animation

Equation with pause:

$$f(x) =$$

Here,

2.11 Math Equation Animation

Equation with pause:

$$f(x) = x^2 + 2x + 1$$

$$=$$

Here, we have the expression of $f(x)$.

2.11 Math Equation Animation

Equation with pause:

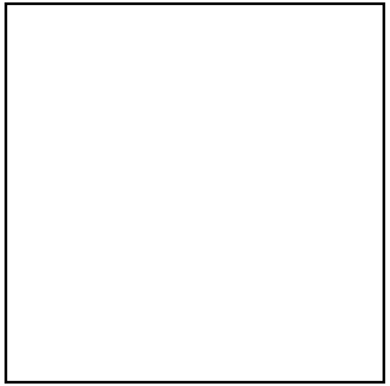
$$\begin{aligned} f(x) &= x^2 + 2x + 1 \\ &= (x+1)^2 \end{aligned}$$

Here, we have the expression of $f(x)$.

By factorizing, we can obtain this result.

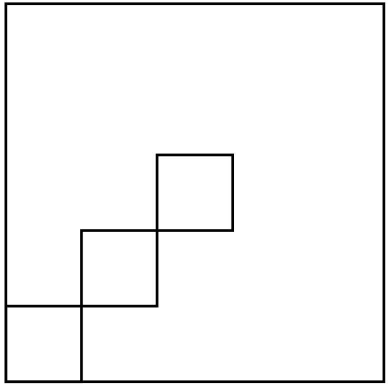
2.12 CeTZ Animation

CeTZ Animation in Touying:



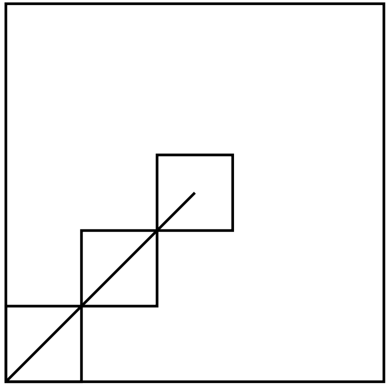
2.12 CeTZ Animation

CeTZ Animation in Touying:



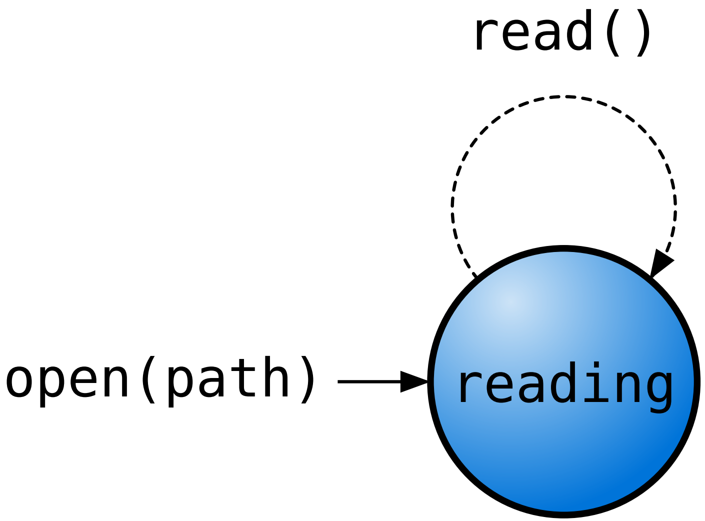
2.12 CeTZ Animation

CeTZ Animation in Touying:



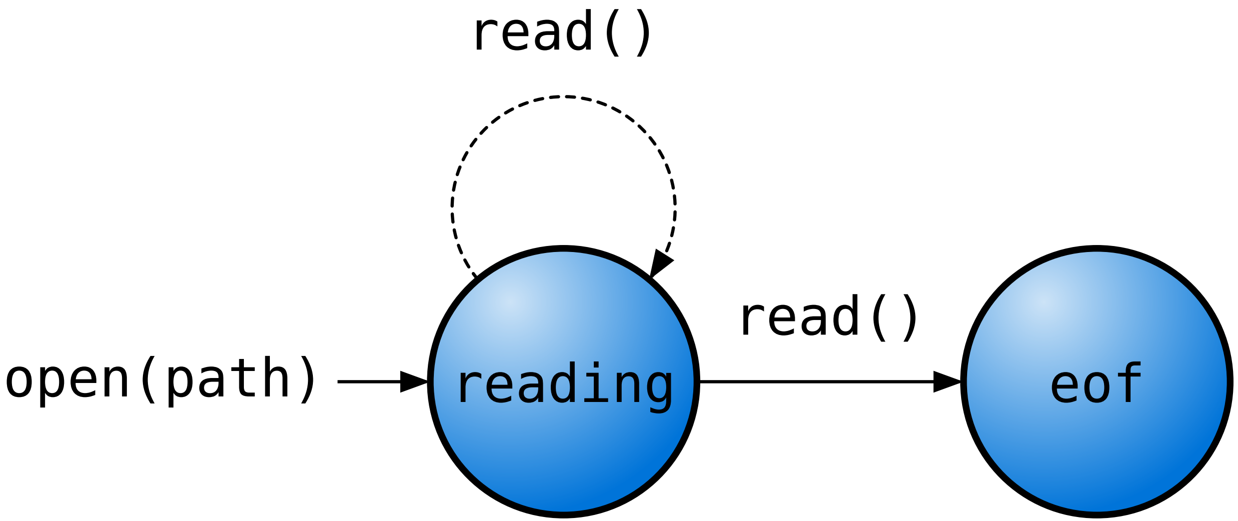
2.13 Fletcher Animation

Fletcher Animation in Touying:



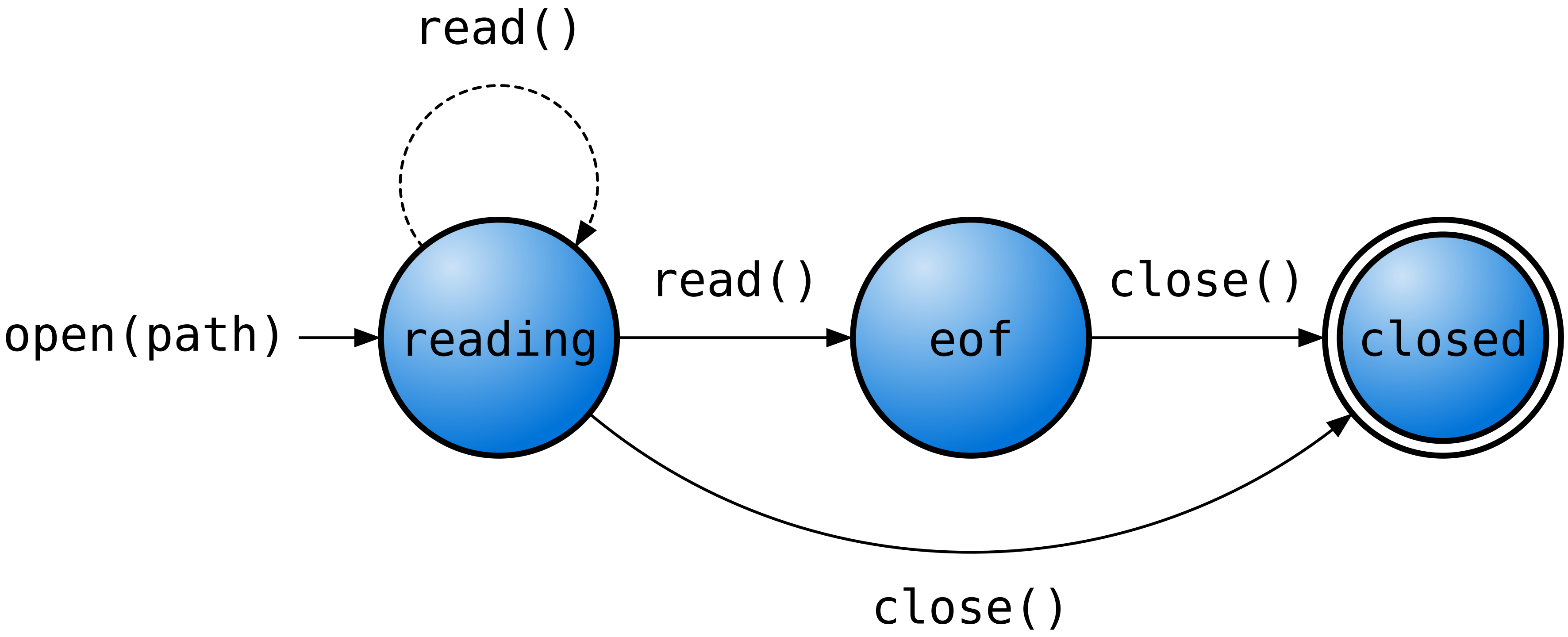
2.13 Fletcher Animation

Fletcher Animation in Touying:



2.13 Fletcher Animation

Fletcher Animation in Touying:



3. Theorems

3.1 Prime numbers

Definition 3.1.1 A natural number is called a *prime number* if it is greater than 1 and cannot be written as the product of two smaller natural numbers.

Example. The numbers 2, 3, and 17 are prime. Corollary 3.1.2.1 shows that this list is not exhaustive!

Theorem 3.1.2 (Euclid) There are infinitely many primes.

3.1 Prime numbers

Proof. Suppose to the contrary that $p_1, p_2, \ldots, p_n$ is a finite enumeration of all primes. Set $P = p_1 p_2 \ldots p_n$. Since $P + 1$ is not in our list, it cannot be prime. Thus, some prime factor p_j divides $P + 1$. Since p_j also divides P, it must divide the difference $(P + 1) - P = 1$, a contradiction. □

Corollary 3.1.2.1 There is no largest prime number.

Corollary 3.1.2.2 There are infinitely many composite numbers.

3.1 Prime numbers

Theorem 3.1.3 There are arbitrarily long stretches of composite numbers.

Proof. For any $n > 2$, consider

$$n! + 2, \quad n! + 3, \quad \ldots, \quad n! + n$$

$\square$

4. Others

4.1 Side-by-side

First column.

Second column.

4.2 Multiple Pages

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum,

4.2 Multiple Pages

quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

5. Appendix

5.1 Appendix

Please pay attention to the current slide number.