property P: G is a comparability graph.

property $\overline{P}$: $\overline{G}$ is a comparability graph.

property C: G is a chordal graph.

property $\overline{C}$: $\overline{G}$ is a chordal graph.

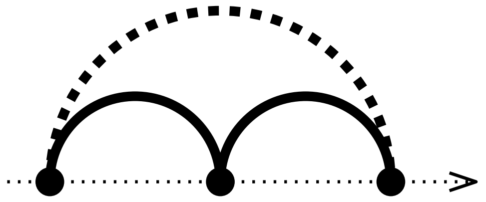
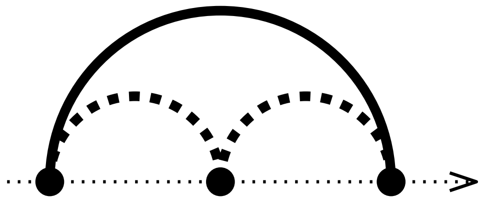
P	$\overline{P}$	C	$\overline{C}$	graph class	
✓				comparability graphs	Chap.4
		✓		chordal graphs	Chap.3
	✓	✓		interval graphs	Chap.7
		✓	✓	split graphs	Chap.5
✓	✓			permutation graphs	Chap.6
✓		✓		cycle-free partial orders	???

Thm.

For every (undirected) graph $G = (V, E)$ the following are equivalent:

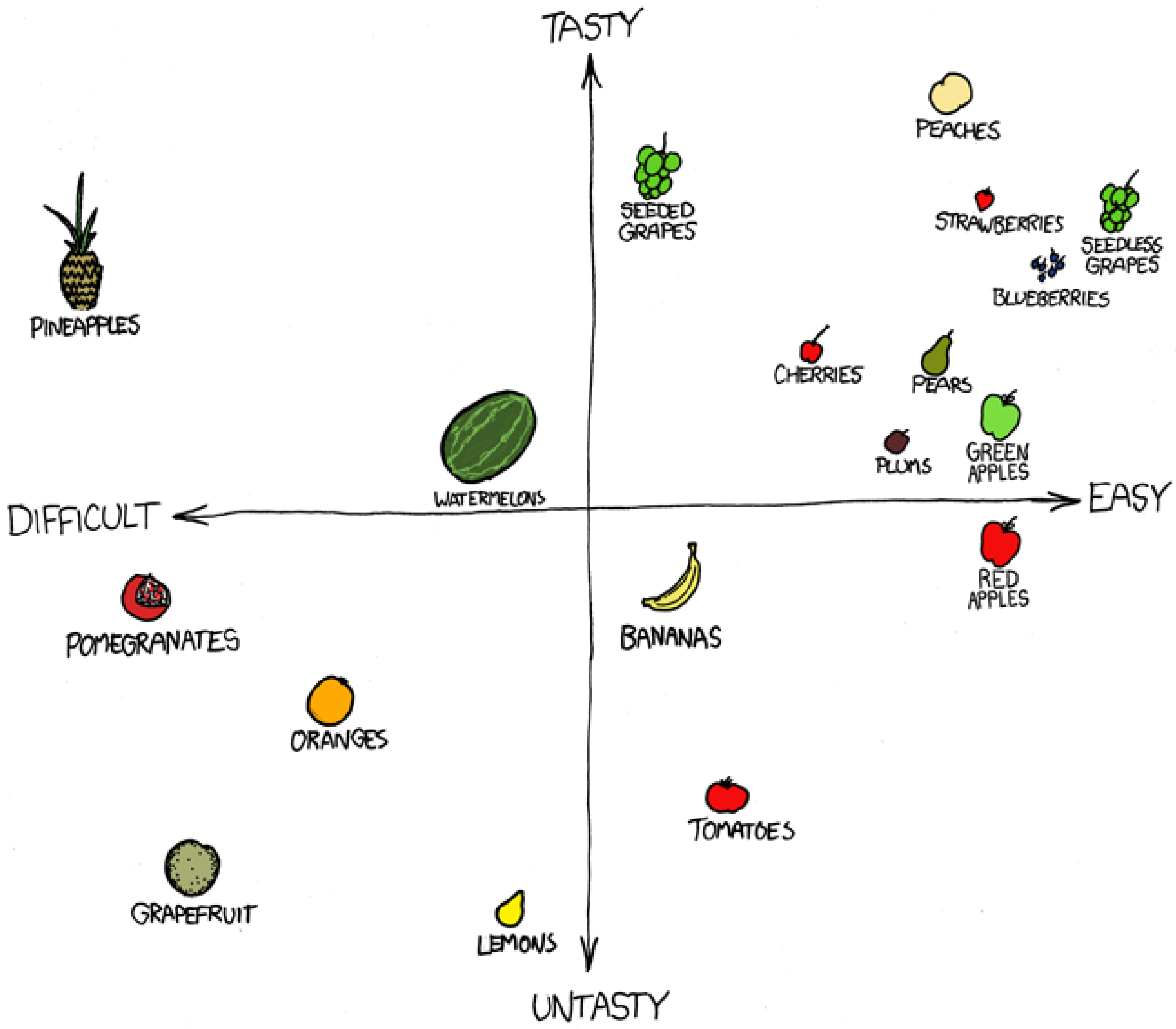
(i) G and $\bar{G}$ are comparability graphs.

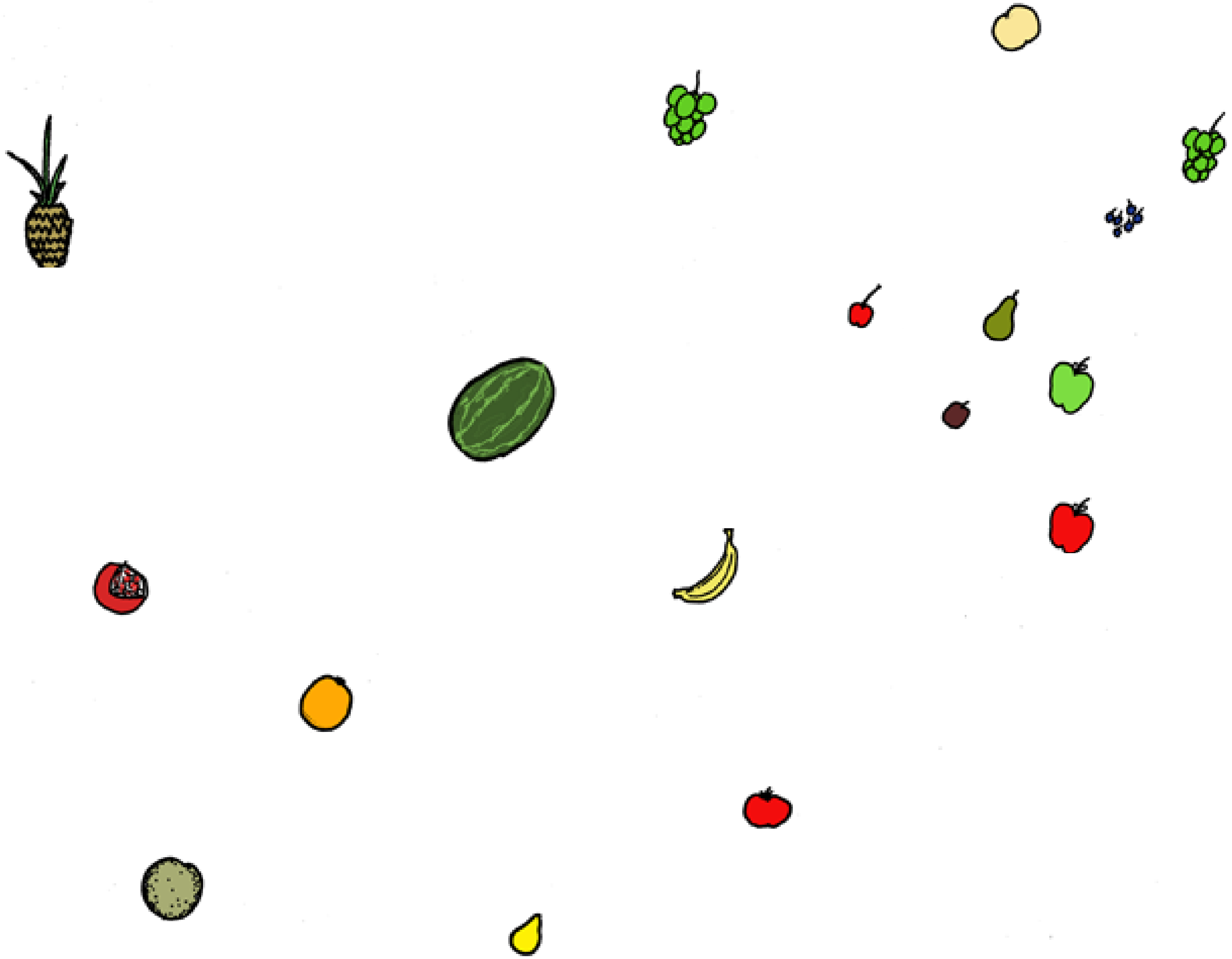
(ii) There exists a vertex ordering σ of G

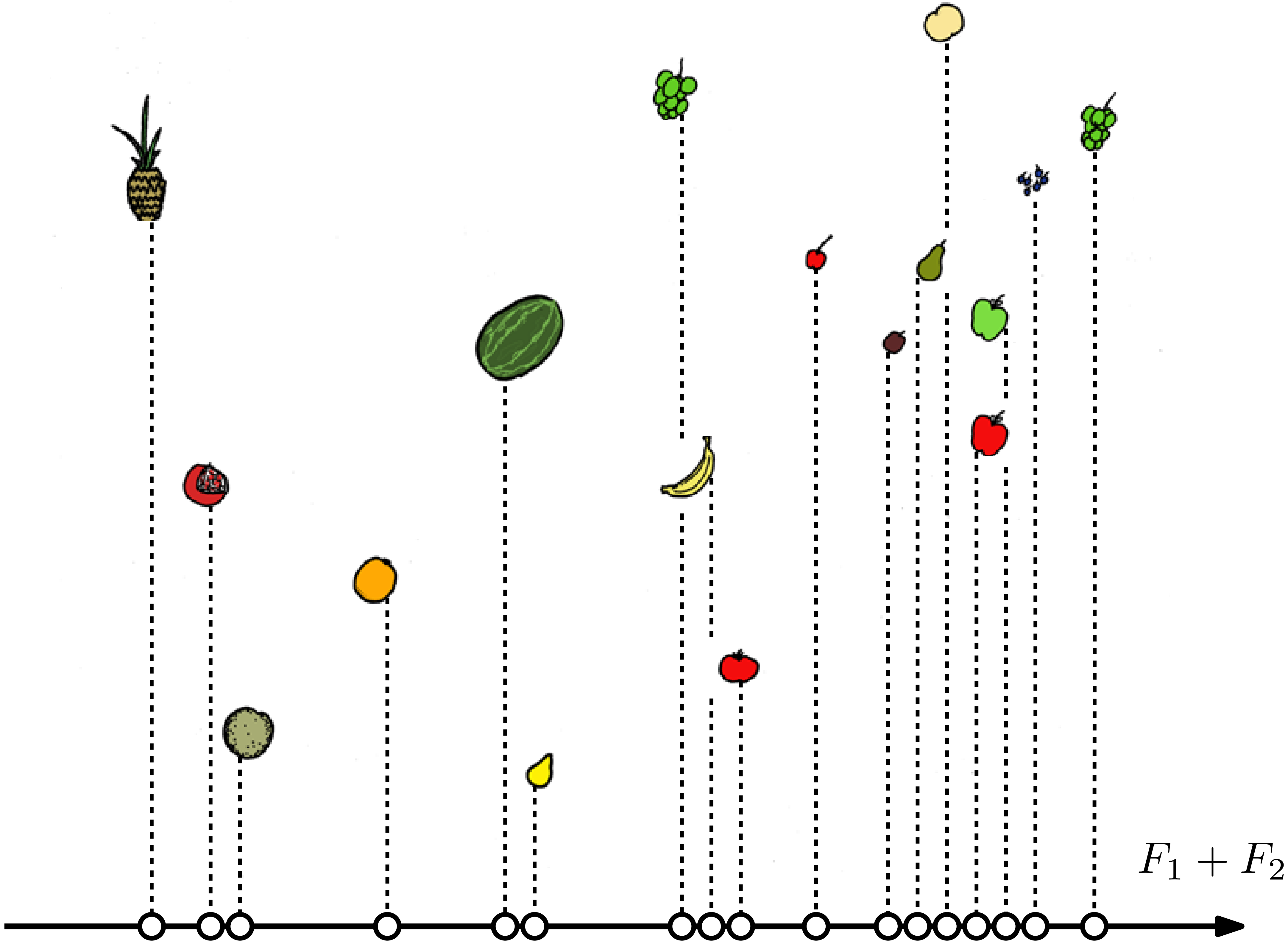
without  and without 

(iii) There exists an *embedding* $V \to \mathbb{R}^2$ such that

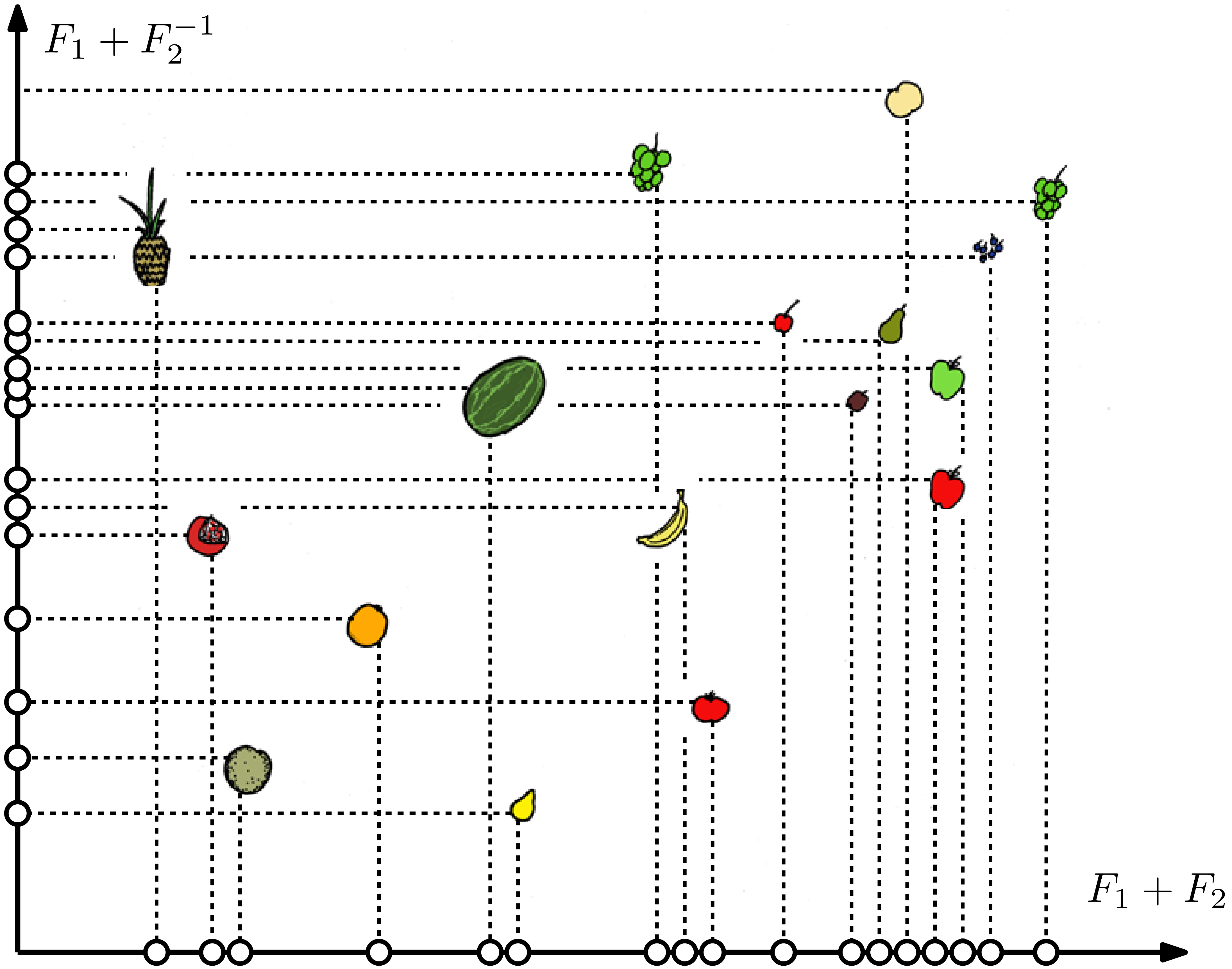
$uv \in E$ if and only if $u_x < v_x \Leftrightarrow u_y < v_y$

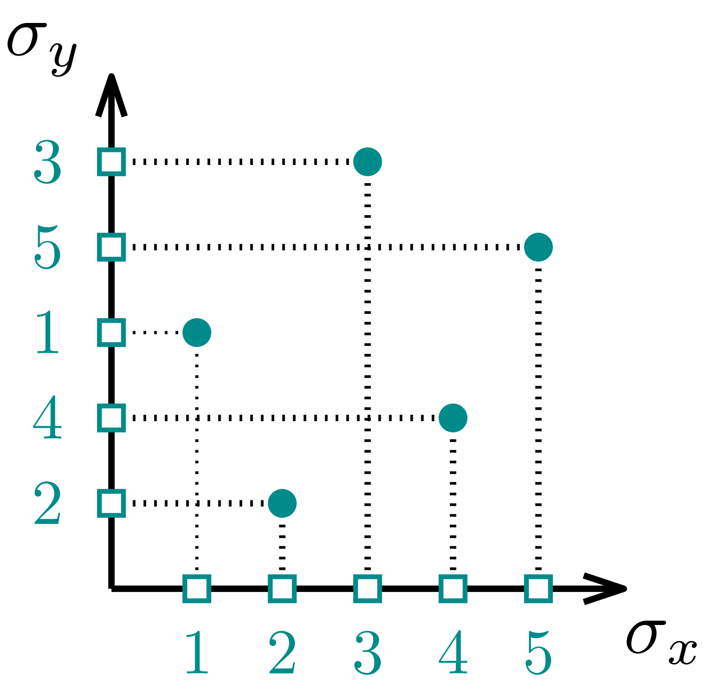

TASTY
UNTASTY
DIFFICULT
EASY
PEACHES
SEEDED GRAPES
STRAWBERRIES
SEEDLESS GRAPES
BLUEBERRIES
PINEAPPLES
CHERRIES
PEARS
WATERMELONS
PLUMS
GREEN APPLES
RED APPLES
POMEGRANATES
BANANAS
ORANGES
TOMATOES
GRAPEFRUIT
LEMONS



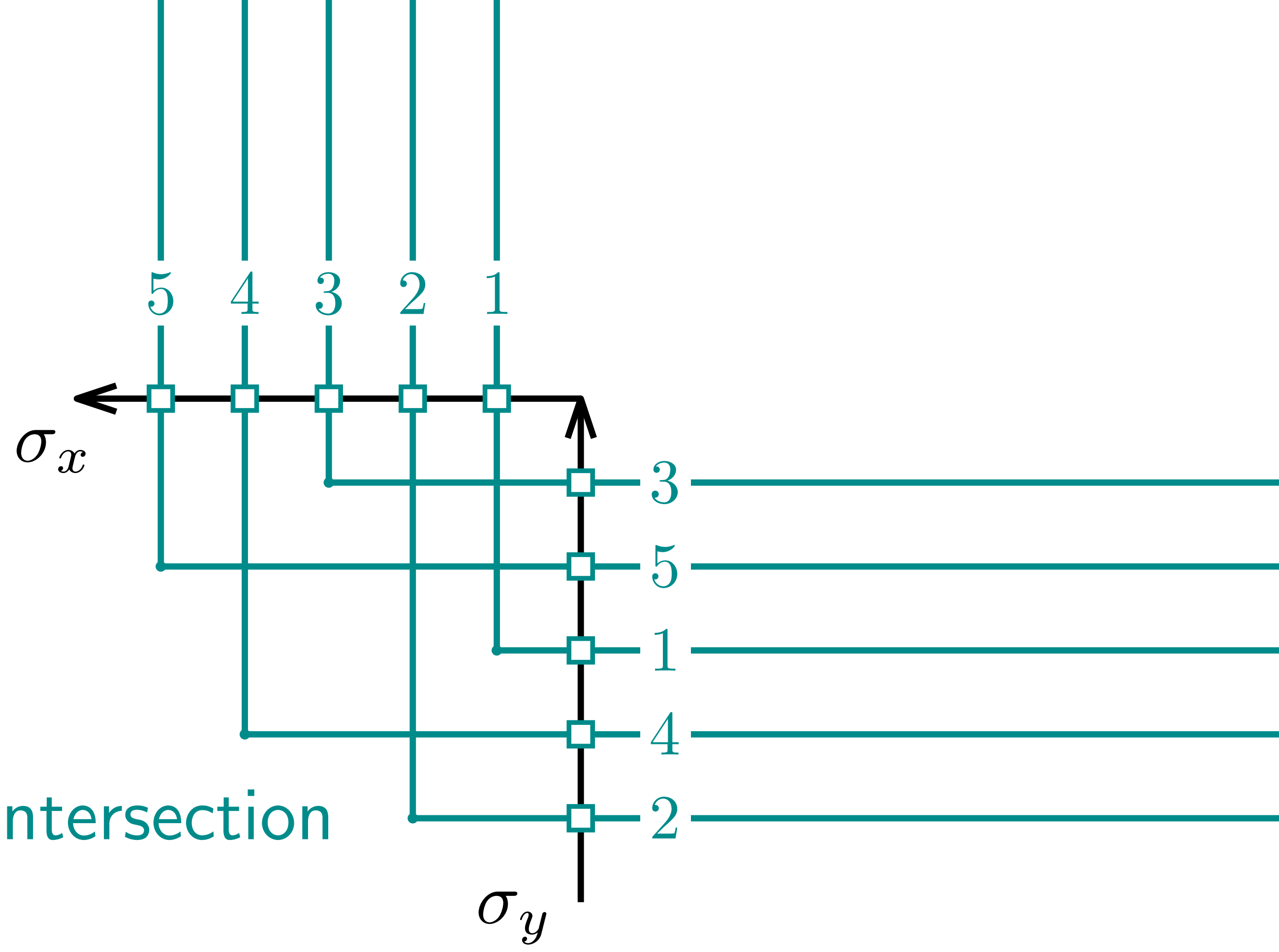


$F_1 + F_2$

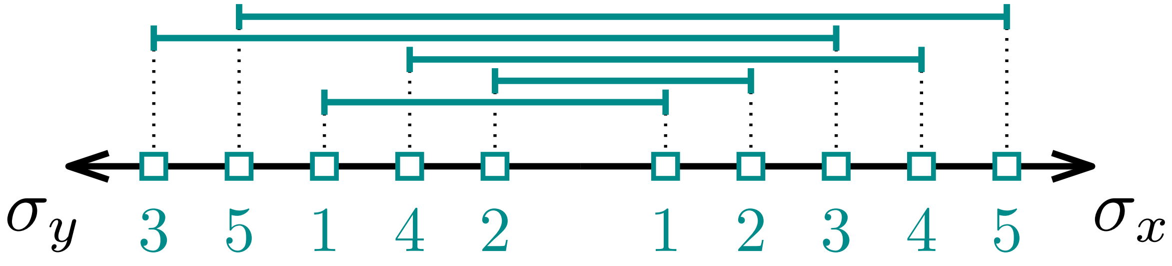

$F_1 + F_2^{-1}$
$F_1 + F_2$



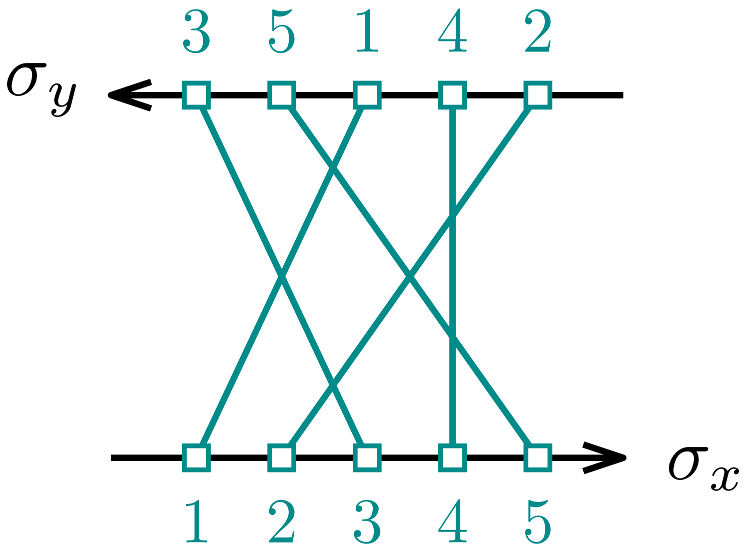
points, dominance



L's, intersection



intervals, containment



segments, intersection

property P**:** G is a comparability graph.

property $\overline{P}$**:** $\overline{G}$ is a comparability graph.

property C**:** G is a chordal graph.

property $\overline{C}$**:** $\overline{G}$ is a chordal graph.

P	$\overline{P}$	C	$\overline{C}$	graph class	
✓				comparability graphs	Chap.4
		✓		chordal graphs	Chap.3
	✓	✓		interval graphs	Chap.7
		✓	✓	split graphs	Chap.5
✓	✓			permutation graphs	Chap.6
✓		✓		cycle-free partial orders	???

Thm 7.1.

For every graph $G = (V, E)$ the following are equivalent:

(i) G is an interval graph.

(ii) $\exists$ vertex ordering σ without 

(iii) G is chordal and $\overline{G}$ is a comparability graph.

(iv) $C_4 \not\subseteq_{\text{ind}} G$ and $\overline{G}$ is a comparability graph.

(v) There exists an ordering $A_1, \ldots, A_x$ of the inclusion-maximal cliques in G such that

$\forall v \in V$ the numbers in $\{i \mid v \in A_i\}$
are consecutive in $\{1, \ldots, x\}$.

Input : undirected graph $G = (V, E)$.
Output : $\textsc{Yes}$ if G is interval graph, $\textsc{No}$ otherwise.

Compute σ_1 with LexBFS;

Algorithm 9 : Recognition of Interval Graphs

Input : undirected graph $G = (V, E)$.
Output : $\textsc{Yes}$ if G is interval graph, $\textsc{No}$ otherwise.

1 Compute σ_1 with LexBFS;
2 Compute σ_2 with LexBFS using σ_1 as tie breaker;

Algorithm 9 : Recognition of Interval Graphs

Input : undirected graph $G = (V, E)$.
Output : YES if G is interval graph, NO otherwise.

1. Compute σ_1 with LexBFS;
2. Compute σ_2 with LexBFS using σ_1 as tie breaker;
3. Compute σ_3 with LexBFS using σ_2 as tie breaker;

Algorithm 9 : Recognition of Interval Graphs

Input : undirected graph $G = (V, E)$.
Output : $\textsc{Yes}$ if G is interval graph, $\textsc{No}$ otherwise.

```
Compute σ1 with LexBFS;
Compute σ2 with LexBFS using σ1 as tie breaker;
Compute σ3 with LexBFS using σ2 as tie breaker;
Compute σ4 with LexBFS using σ3 as tie breaker;
```

Algorithm 9 : Recognition of Interval Graphs

Input : undirected graph $G = (V, E)$.
Output : $\textsc{Yes}$ if G is interval graph, $\textsc{No}$ otherwise.

Compute σ_1 with LexBFS;
Compute σ_2 with LexBFS using σ_1 as tie breaker;
Compute σ_3 with LexBFS using σ_2 as tie breaker;
Compute σ_4 with LexBFS using σ_3 as tie breaker;
Return whether σ_4 contains forbidden pattern;

Algorithm 9 : Recognition of Interval Graphs