Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea in Matematica

Recognition of Collapsible Complexes and its NP-completeness

Tesi di Laurea in Geometria Computazionale

Relatore:
Chiar.mo Prof.
Giovanni Paolini

Presentata da:
Giuseppe Aristodemo

Anno Accademico 2023/2024

loading...

Introduzione

Dato uno spazio topologico, è naturale chiedersi se esso sia contraibile; formalmente, si definisce il problema della contraibilità e se ne studia la decidibilità. Questa è una domanda a cui non è ovvio poter rispondere, poiché i concetti appena menzionati sono molto astratti. Ci si restringe quindi a una classe più "concreta" di spazi, i complessi simpliciali, e si definisce la collassabilità, una proprietà che implica la precedente, ma che può essere meglio studiata da un punto di vista combinatorico.
Questa tesi si focalizza sullo studio della collassabilità di complessi simpliciali, analizzandone inoltre la complessità computazionale. La trattazione è suddivisa in quattro capitoli più un'appendice.

Il primo capitolo fornisce i preliminari necessari in seguito. Più precisamente, vengono prima introdotti i poset, ossia insiemi parzialmente ordinati; successivamente vengono definiti i complessi simpliciali astratti e se ne analizzano le proprietà.

Nel secondo capitolo viene introdotta la Teoria di Morse Discreta; in particolare, si definisce il concetto di collassamento simpliciale di un complesso e si studiano condizioni che permettano di valutare quando e come un complesso sia collassabile a un suo sottocomplesso. I concetti chiave sono quelli di matching aciclico e oggetto universale; il primo è una particolare relazione sul diagramma di Hasse del complesso e il secondo è un poset che, in un certo senso, ne descrive tutte le sequenze possibili di collassamenti. Il risultato più importante in questo capitolo è il Patchwork Theorem, strumento che ci darà un modo per costruire più agevolmente matching aciclici. Infine dimostreremo tre risultati notevoli che daranno anche un'idea di come la Teoria di Morse viene applicata.

Dedicato all'introduzione di concetti necessari dell'informatica teorica, il terzo capitolo tratta la complessità computazionale; in particolare vedremo come un problema viene codificato e descritto in termini formali, e come viene descritta la difficoltà nel risolverlo. A seguire approfondiremo quest'ultimo concetto introducendo le riduzioni ed esse ci permetteranno di dimostrare quando, in un certo senso, un problema è più o meno difficile di un altro. Ciò porterà in modo naturale alla definizione di NP-completezza e vedremo alcuni esempi di problemi notevoli.

Il quarto e ultimo capitolo tratta la parte centrale della tesi, ovvero il problema del riconoscere quando un complesso simpliciale di dimensione d sia collassabile o meno ad un suo sottocomplesso di dimensione k; ciò si riassume in (d, k)-Collapsibility. In particolare, seguiremo l'approccio di Malgouyres e Francés ripercorrendone la dimostrazione dell'NP-completezza del problema nel caso in cui $d = 3$ e $k = 0$; in altre parole, è NP-completo decidere se un complesso di dimensione 3 sia collassabile ad un punto. In seguito vedremo come questo risultato può essere esteso e citeremo risultati analoghi al precedente, in particolar modo quelli di Tancer. Chiude il problema un articolo di Paolini, in cui viene dimostrata l'NP-completezza nei casi rimanenti.

In conclusione, il risultato finale è l'NP-completezza di (d, k)-Collapsibility per $d \geq k + 2$, eccetto il caso di $(2, 0)$-Collapsibility, che si mostra essere risolvibile in tempo polinomiale. La decidibilità in tempo polinomiale vale anche per il caso rimanente, $(k + 1, k)$-Collapsibility.

In appendice si trova la dimostrazione dell'NP-completezza di un problema notevole, CIRCUIT-SAT, e uno sketch di dimostrazione dell'indecidibilità per quanto riguarda il problema della contraibilità.

Contents

Chapter 1

Preliminaries

1.1 Posets

Definition 1.1. *A partially ordered set (or **poset**, for short) $(P, \leq)$ is a set P together with a binary relation $\leq$, satisfying the following axioms:*

(i) for all $t \in P$, we have $t \leq t$ (reflexivity);

(ii) if $s \leq t$ and $t \leq s$, then $s = t$ (antisymmetry);

(iii) if $s \leq t$ and $t \leq u$, then $s \leq u$ (transitivity).

We say that two elements s and t of P are *comparable* if $s \leq t$ or $t \leq s$; otherwise s and t are *incomparable*, denoted by $s \,||\, t$. A *chain* is a poset in which any two elements are comparable.
We say that P has a $\hat{0}$ if there exists an element $\hat{0} \in P$ such that $\hat{0} \leq t$ for all $t \in P$. Similarly, P has a $\hat{1}$ if there exists $\hat{1} \in P$ such that $t \leq \hat{1}$ for all $t \in P$. We denote by $\hat{P}$ the poset obtained from P by adjoining a $\hat{0}$ and $\hat{1}$. On the other hand, we denote with $\overline{P}$ the poset obtained by removing $\hat{0}$ and $\hat{1}$, when they exist.

Example 1.2.

Let $n \in \mathbb{N} \setminus \{0\}$. The set $[n]$[1] with its usual order forms a poset with the property that any two elements are comparable.

An *order-preserving* map (also called a poset map) between two posets P and Q is $\phi : P \to Q$ with the following property:

$$s \leq t \text{ in } P \iff \phi(s) \leq \phi(t) \text{ in } Q.$$

Such a map is called an *isomorphism* between P and Q if it is a bijection and its inverse is also order-preserving.

[1] With this symbol we simply denote the set $\{1, 2, \ldots, n\}$.

Lattices

An important class of posets are lattices. If P is a poset and $s, t \in P$, then an *upper bound* of s and t is an element $u \in P$ such that $s, t \leq u$. A *join* is a least upper bound, that is, an upper bound u of s and t satisfying $u \leq v$ for any other upper bound v. If a least upper bound of s and t exists, then it is clearly unique and is denoted $s \vee t$. Similarly we define the *meet* $s \wedge t$, that is, the greatest lower bound, when it exists.
A **lattice** L is a poset for which every pair of elements has a join and a meet. Thus we can see $\vee$ and $\wedge$ as operations and in a lattice:

- the above operations are associative, commutative, and idempotent;
- $s \wedge (s \vee t) = s = s \vee (s \wedge t)$ (absorption laws); $s \wedge t = s \iff s \vee t = t \iff s \leq t$.

Clearly all finite lattices have a $\hat{0}$ and $\hat{1}$.

1.2 Abstract Simplicial Complexes

Definition 1.3. *A finite **abstract simplicial complex** is a finite set A together with a collection Δ of subsets of A such that if $X \in \Delta$ and $Y \subseteq X$, then $Y \in \Delta$.*

We denote the abstract simplicial complex simply by Δ. An element $v \in A$ such that $\{v\} \in \Delta$ is called a *vertex*. We denote the set of all vertices of Δ by $V(\Delta)$. When Δ consists of all subsets of A, it is called a *simplex*, and is denoted by Δ^A. The sets $\sigma \in \Delta$ are called *simplices*, and those that are contained in no other simplex of Δ are called *maximal*.

Given two finite abstract simplicial complexes Δ_1 and Δ_2 such that $\sigma \in \Delta_1$ implies $\sigma \in \Delta_2$, we say that Δ_1 is an abstract simplicial *subcomplex* of Δ_2, written $\Delta_1 \subseteq \Delta_2$.
We also define the **dimension** of each simplex, which is 1 less than its cardinality as a set. The dimension of a finite abstract simplicial complex is said to be the maximum of the dimensions of its simplices, denoted by $\dim \Delta$.
Clearly, $\Delta_1 \subseteq \Delta_2$ implies $\dim \Delta_1 \leq \dim \Delta_2$.

Definition 1.4. *Let Δ_1 and Δ_2 be two finite abstract simplicial complexes. A **simplicial map** from Δ_1 to Δ_2 is a map $f : V(\Delta_1) \to V(\Delta_2)$ such that $\sigma \in \Delta_1$ implies $f(\sigma) \in \Delta_2$. We simply write $f : \Delta_1 \to \Delta_2$.*

Remark 1.5.

- The identity map is a simplicial map from an abstract simplicial complex onto itself.

- If $\Delta_1 \subseteq \Delta_2$, we have a natural simplicial inclusion map.
- The composition of two simplicial maps is again a simplicial map.
- If f is bijective and simplicial, its inverse is not necessarily simplicial.
- If A is a finite set, any $f : V(\Delta) \to A$ induces a simplicial map $f : \Delta \to \Delta^A$.

Definition 1.6. *Let Δ_1 and Δ_2 be abstract simplicial complexes, and let $f : \Delta_1 \to \Delta_2$ be a simplicial map. Then f is called an **isomorphism** of abstract simplicial complexes if it is a bijection and its inverse is also a simplicial map. In this case we say that the two complexes are isomorphic.*

An important class of simplicial maps are isomorphisms $f : \Delta \to \Delta$; these maps are called **automorphisms** of Δ and they forms a group denoted by $\mathrm{Aut}(\Delta)$.

We associate a standard combinatorial gadget to an abstract simplicial complex: its face poset.

Definition 1.7. *The **face poset** of Δ is denoted $\mathcal{F}(\Delta)$ and it is the poset consisting of all nonempty simplices of Δ, with the partial order induced by the inclusion relation on the set of simplices.*

Recall that for a poset $(P, <)$, a **linear extension** L is a total order $<_L$ on the set of elements of P such that for any $x, y \in P$, we have $x <_L y$ whenever $x < y$; clearly, $(P, <_L)$ is a chain.
We can also think of a linear extension as a poset map $L : (P, <) \to (P, <_L)$, where $L : P \to P$ is simply the identity map.

Example 1.8. For an arbitrary abstract simplicial complex Δ, a standard linear extension of the face poset $\mathcal{F}(\Delta)$ is obtained by setting $\sigma <_L \tau$ whenever $\dim \sigma < \dim \tau$, and choosing an arbitrary order for each set of simplices of the same dimension.

Simplicial Join

The following is a classical construction that produces new abstract simplicial complexes from old ones.

Definition 1.9. *Let Δ_1 and Δ_2 be two abstract simplicial complexes whose vertices are indexed by disjoint sets. The **join** of Δ_1 and Δ_2 is the abstract simplicial complex $\Delta_1 * \Delta_2$ is $V(\Delta_1) \cup V(\Delta_2)$, and the set of simplices is given by*

$$\Delta_1 * \Delta_2 = \{\sigma \subseteq V(\Delta_1) \cup V(\Delta_2) \mid \sigma \cap V(\Delta_1) \in \Delta_1 \text{ and } \sigma \cap V(\Delta_2) \in \Delta_2\}.$$

Clearly, we have commutativity: the joins $\Delta_1 * \Delta_2$ and $\Delta_2 * \Delta_1$ are isomorphic. In this sense, we have also associativity.

Joining with the abstract simplicial complex consisting of a single vertex is also called *coning*. We can also take a join with the abstract simplicial complex with n vertices and no simplices of dimension 1 and higher; this is called the *n-coning*.

1.3 Polyhedral Complexes

Given an abstract simplicial complex, we would like to associate a topological space to it. In order to accomplish this, we need to define some geometric objects.

Let A be a set of $n+1$ affine independent points in $\mathbb{R}^N$, with $N \geq n$. A **geometric n-simplex** σ is the convex hull of A. The convex hull of the subsets of A are called **subsimplices** of σ.
The **standard** n-simplex is the convex hull of the standard unit basis in $\mathbb{R}^{n+1}$.

More generally, given a finite set A, we have the vector space $\mathbb{R}^A$, whose coordinates are indexed by the elements of A; correspondingly, for any subset $B \subseteq A$, we can define the standard B-simplex in $\mathbb{R}^A$ as the one spanned by the subset of the standard unit basis indexed by elements in B.

Definition 1.10. *Given a finite abstract simplicial complex Δ, we define its **standard geometric realization** to be the topological space obtained by taking the union of standard σ-simplices in $\mathbb{R}^{V(\Delta)}$, for all $\sigma \in \Delta$.*
*Any topological space homeomorphic to the standard geometric realization of Δ is called its **geometric realization**, and it is denoted by $|\Delta|$. However, with an abuse of notation, most of the time we will simply denote it Δ.*

Recall that a *convex polytope* P is a bounded subset of $\mathbb{R}^d$ that is the solution of a finite number of linear inequalities and equalities. Equivalently, it is the convex hull of a finite set of points.
A subset $F \subseteq P$ is called a *face* of P if there exists a linear function f on $\mathbb{R}^d$ such that $f(x) = 0$, for all $x \in F$, and $f(x) \geq 0$, for all $x \in P$.

Definition 1.11. *A **geometric polyhedral complex** Γ in $\mathbb{R}^n$ is a collection of convex polytopes in $\mathbb{R}^n$ such that:*

(i) every face of a polytope in Γ is itself a polytope in Γ;

(ii) the intersection of any two polytopes in Γ is a face of each of them.

Most of the terminology carries over from the simplicial context. One important property worth observing is that a direct product of two geometric polyhedral complexes is again a geometric polyhedral complex.

Let us now define a more general family of complexes:

1. We start with a discrete abstract set of points; this is called the *0-skeleton*. Then, we proceed by induction on the dimension of the attached faces.

2. At step d we attach the d-dimensional faces.
   Each face is represented by some convex polytope P in $\mathbb{R}^d$. To attach it we need a continuous map $f : \partial P \to X$, where X denotes the part of the complex constructed in the first $d-1$ steps. The attaching map should induce a homeomorphism $f|_{\partial P} : \partial P \to f(\partial P)$, and this homeomorphism should preserve the cell structure; the cell structure on ∂P is the polytopal structure, and the one on $f(\partial P)$ is induced from the gluing process.

When a cell is obtained by gluing the polyhedron P on the complex, we simply say that the cell is P.
When a topological space can be obtained by the above gluing procedure it is called a **polyhedral complex**.

Actually, we will only discuss about **generalized simplicial complexes**, which are polyhedral complexes with simplices as cells.

Geometry of the Simplicial Join

One can define the join of arbitrary topological spaces. Let I denote the closed unit interval $[0, 1]$.

Definition 1.12. *Let X and Y be two topological spaces. The **join** of X and Y is the topological space $X * Y$ defined as follows:*

$$X * Y = I \times X \times Y \Big/ \sim$$

where the equivalence relation $\sim$ is given by

- $(0, x, y) \sim (0, x, \tilde{y})$*, for all* $y, \tilde{y} \in Y$*;*
- $(1, x, y) \sim (0, \tilde{x}, y)$*, for all* $x, \tilde{x} \in X$*.*

Observe that for two abstract simplicial complexes Δ_1 and Δ_2 we have

$$|\Delta_1| * |\Delta_2| \cong |\Delta_1 * \Delta_2|,$$

where on the right side we take the simplicial join.
In fact, given geometric realizations of Δ_1 in $\mathbb{R}^m$ and Δ_2 in $\mathbb{R}^n$, we can obtain a geometric realization of $\Delta_1 * \Delta_2$ in $\mathbb{R}^{m+n+1}$ as follows. Identify $\mathbb{R}^m$ with the subspace $\{(x_1, \ldots, x_m, 0, \ldots, 0) \in \mathbb{R}^{m+n+1} \mid (x_1, \ldots, x_m) \in \mathbb{R}^m\}$, and identify $\mathbb{R}^n$ with the subspace $\{(0, \ldots, 0, y_1, \ldots, y_n, 1) \in \mathbb{R}^{m+n+1} \mid (y_1, \ldots, y_n) \in \mathbb{R}^n\}$. Take now the induced embeddings of $|\Delta_1|$ and $|\Delta_2|$ into $\mathbb{R}^{m+n+1}$ and let $|\Delta_1 * \Delta_2|$ be the union of convex hulls of pairs of simplices: one from Δ_1 and one from Δ_2.

Example 1.13. Let m and n be natural numbers. Then we have

$$\mathbb{S}^m * \mathbb{S}^n \cong \mathbb{S}^{m+n+1}.$$

To see this, note that in general the join is associative, that is, $(X * Y) * Z \cong X * (Y * Z)$. Therefore

$$\mathbb{S}^m * \mathbb{S}^n \cong \underbrace{\mathbb{S}^0 * \cdots * \mathbb{S}^0}_{m+1} * \underbrace{\mathbb{S}^0 * \cdots * \mathbb{S}^0}_{n+1} \cong \underbrace{\mathbb{S}^0 * \cdots * \mathbb{S}^0}_{m+n+2} \cong \mathbb{S}^{m+n+1}.$$

1.4 Examples

Simplicial Flag Complexes

For a graph G and a subset $S \subseteq V(G)$ of its vertices, we let $G[S]$ denote the corresponding subgraph, and if it is complete we call it a *clique*.

Definition 1.14. *Given an arbitrary graph G, we let $\mathrm{Cl}(G)$ denote the abstract simplicial complex whose set of vertices is $V(G)$ and whose simplices are all subsets $S \subseteq V(G)$ such that $G[S]$ is a complete graph.*

The abstract simplicial complex $\mathrm{Cl}(G)$ is called a **flag complex** in algebraic topology, while it is called a *clique complex* in combinatorics.

Given a graph G, a set of vertices $S \subseteq V(G)$ is called **independent** if for all $v, w \in S$ we have $(v, w) \notin E(G)$.

Definition 1.15. *For an arbitrary graph G, the **independence complex** of G, denoted $\mathrm{Ind}(G)$, is the abstract simplicial complex whose set of vertices is $V(G)$ and whose simplices are all the independent sets of G.*

Remark 1.16. Given a graph G, we denote with $\overline{G}$ the *complement graph*, that is, a graph on the same vertices such that $(u, v) \in E(\overline{G})$ if and only if $(u, v) \notin E(G)$.
Since independent sets of G are the same as the cliques of $\overline{G}$, we see that $\mathrm{Ind}(G)$ is isomorphic to $\mathrm{Cl}(\overline{G})$ as abstract simplicial complexes.

Order complexes

Another classical example is the order complex of a poset.

Definition 1.17. *Let P be a poset. Define the* ***order complex*** $\Delta(P)$ *to be the abstract simplicial complex whose vertices are all elements of P and whose simplices are all finite chains of P, including the empty one.*

Example 1.18.

- If A is a totally ordered, finite set, then $\Delta(A) = \Delta^A$.
- Let $O_n = \{a_1^1, \ldots, a_n^1, a_1^2, \ldots, a_n^2\}$, with the partial order generated by $a_i^p > a_{i+1}^q$, for all $p, q \in \{1, 2\}, i = 1, \ldots, n-1$. Then the order complex of O_n is the join of n copies of $\mathbb{S}^0$. This can be realized as the boundary of a polytope called a *cross-polytope*; in particular, it is homeomorphic to $\mathbb{S}^{n-1}$.
- Let $n \in \mathbb{N}$, and let $\mathcal{B}_n$ be the set of all subsets of $[n]$, partially ordered by inclusion, called the boolean lattice. One can see that $\Delta(\overline{\mathcal{B}}_n)$ is isomorphic to the barycentric subdivision of the boundary of an $(n-1)$-simplex; in particular, it is homeomorphic to $\mathbb{S}^{n-2}$.
- Let $n \in \mathbb{N}$. The *partition lattice* Π_n is the poset whose elements are all the set partitions of $[n]$, and the partial order is that of refinement. The partition lattice has a minimal element $(1)(2)\ldots(n)$ and a maximal element $[n]$. We will show that $\Delta(\overline{\Pi}_n)$ is homotopy equivalent to a wedge of $(n-1)!$ copies of $\mathbb{S}^{n-3}$.

Chapter 2

Discrete Morse Theory

Let Δ be a generalized simplicial complex, and let σ and τ be simplices of Δ such that:

(i) $\tau \subset \sigma$, in particular $\dim \tau < \dim \sigma$;

(ii) σ is a maximal simplex, and no other maximal simplex contains τ.

We call a **simplicial collapse** of Δ the removal of all simplices γ such that $\tau \subseteq \gamma \subseteq \sigma$. If additionally, we have $\dim \tau = \dim \sigma - 1$, then this is called an **elementary collapse**. When for two generalized simplicial complexes Δ_1 and Δ_2 there exists a sequence of collapses leading from Δ_1 to Δ_2, we write $\Delta_1 \searrow \Delta_2$.

Proposition 2.1. *A sequence of collapses induces a strong deformation retraction, and in particular, a homotopy equivalence.*

Remark 2.2. Note that an elementary collapse is possible if and only if there exists a simplex τ whose link in Δ consists of a single vertex v; the simplex σ is then given by the span of σ and v.

The combinatorial encoding of a set of collapses is best provided by a matching consisting of a collection of pairs of cells (τ, σ) such that $\tau \subset \sigma$, and $\dim \sigma = \dim \tau + 1$. Morse theory examines when such a matching can be turned into a *sequence of collapses*.

2.1 Acyclic Matchings

Definition 2.3. *A **partial matching** in a poset $(P, <)$ is a partial matching in the underlying graph of its Hasse diagram, i.e., it is a subset $M \subseteq P \times P$ such that:*

- *$(a, b) \in M$ implies $b \succ a$;*

- *each $a \in P$ belongs to at most one element in M.*

When $(a, b) \in M$, we write $a = d(b)$ and $b = u(a)$.
A partial matching on P is said to be ***acyclic*** *if there is no cycle*

$$b_1 \succ d(b_1) \prec b_2 \succ d(b_2) \prec \cdots \prec b_n \succ d(b_n) \prec b_1 \tag{2.1}$$

with $n \geq 2$, and all $b_i \in P$ being distinct.

A popular, more intuitive way to reformulate the acyclicity condition is the following. Given a poset P, we can orient all edges in its Hasse diagram so that they point from the larger element to the smaller one. After that, given a partial matching M, change the orientation of the edges in M to the opposite one. Now, the matching is acyclic if and only if the obtained oriented graph has no cycles. Acyclic matchings are also called **Morse matchings**.

Given a partial matching, we can intuitively think about pairs as *internal* collapses. The idea is to remove all the matched elements in some appropriate order, so that the homotopy type of the underlying space remains the same. An unmatched element is said to be **critical**, and we denote the set of critical elements by $C(P, M)$.

From now on we only consider finite posets. We have the following relationship between acyclic matchings and linear extensions.

Theorem 2.4. *A partial matching on P is acyclic if and only if there exists a linear extension L of P such that the elements a and $u(a)$ follow consecutively in L, for all $a \in P$.*

Proof. Assume that we have a linear extension L satisfying the property, and that at the same time, we have a cycle as in (2.1). Set $a_i = d(b_i)$, for $i = 1, \ldots, n$. Then we have

$$b_{i+1} \succ a_i \implies a_i <_L b_{i+1} \implies a_i <_L a_{i+1},$$

since a_{i+1} and b_{i+1} follow consecutively in L. Thus $a_n >_L a_{n-1} >_L \cdots >_L a_1 >_L a_0 = a_n$, yielding a contradiction.

Assume now that we have an acyclic matching, and let us define L inductively. Let Q be the set of elements that are already ordered in L; we start with $Q = \emptyset$. Let W denote the set of minimal elements in $P \setminus Q$. At each step we have one of the following cases.

Case 1. One of the elements c in W is critical.
In this case, we simply add c to the order L as the largest element, and proceed with $Q \cup \{c\}$.

Case 2. All elements in W are matched.
Consider the subgraph of $P \setminus Q$ induced by $W \cup u(W)$. Orient its edges as described above, i.e., they should point from the larger element to the smaller one, except when these two elements are matched, in which case should be oriented in the opposite way. Call this oriented graph G.
If there exists an element $a \in W$ such that the only element in $W \cup u(W)$ smaller than $u(a)$ is a itself, then we can add a and $u(a)$ on top of L, and proceed with $Q \cup \{a, u(a)\}$. Otherwise, observe that the outdegree of $u(a)$ in G is positive, for each $a \in W$. On the other hand, the outdegrees in G of all $a \in W$ equal to 1. Therefore outdegrees of all vertices in G are positive, then we conclude that G must have a cycle, and this contradicts the acyclicity of the matching. □

We would also like to characterize acyclic matchings via a special class of poset maps.

Definition 2.5. *Given two posets P and Q, a poset map $\varphi : P \to Q$ is said to have* ***small fibers*** *if for any $q \in Q$, the fiber $\varphi^{-1}(q)$ is either empty or consists of two comparable elements.*

Remark 2.6. Since φ is a poset map, if for some $q \in Q$ the fiber consists of two elements, then one of these must actually cover the other one.

Therefore, to any poset map with small fibers φ we can associate a partial matching $M(\varphi)$ consisting of all fibers of cardinality 2.

Theorem 2.7. *For any poset map with small fibers $\varphi : P \to Q$, the partial matching $M(\varphi)$ is acyclic.*
Vice versa, any acyclic matching on P can be represented as $M(\varphi)$, for some poset map with small fibers φ.

Proof. Since $\varphi : P \to Q$ is a poset map, then the induced matching $M(\varphi)$ is acyclic: for if it were not, there would exist a cycle as in (2.1), and φ would map this cycle to a set of distinct elements $q_1 > q_2 > \cdots > q_t > q_1$ of Q, for some t, yielding a contradiction.

Conversely, by Theorem 2.4, given an acyclic matching on P, there exists a linear extension L of P such that the elements a and $u(a)$ follow consecutively in L. By gluing them together in this order we define a poset map with small fibers from P to a chain. □

Notice how at some point in the proof of Theorem 2.7, we have defined a poset map with small fibers onto a chain. These maps are especially important, and we give them a separate name.

Definition 2.8. *A poset map with small fibers $\varphi : P \to Q$ is called a* ***collapsing order*** *if it is surjective, and Q is a chain.*

If M is an acyclic matching on P, we say that a collapsing order φ is a collapsing order for M if $M(\varphi) = M$. In this case, the chain Q gives us the order in which we can perform the collapsing sequence.

2.2 Universal Object

In this section we show that for any poset P, and any acyclic matching on it, there exists a *universal object*: a poset whose linear extensions enumerate all allowed collapsing orders.

Definition 2.9. *Let P be a poset, and let M be an acyclic matching on P. We define the* ***universal object*** *$U(P, M)$ to be the poset whose set of elements is $M \cup C(P, M)$, and whose partial order is the transitive closure of the elementary relations given by $S_1 \leq_U S_2$, for $S_1, S_2 \in U(P, M)$ if and only if $x \leq y$, for some $x \in S_1$ and $y \in S_2$.*

Intuitively, we think of elements of M as subsets of P of cardinality 2, while we think of elements of $C(P, M)$ as subsets of P of cardinality 1. Broadly speaking, $U(P, M)$ is obtained from P by gluing each matched pair together to form a single element, then the partial order is that induced by the one in P.

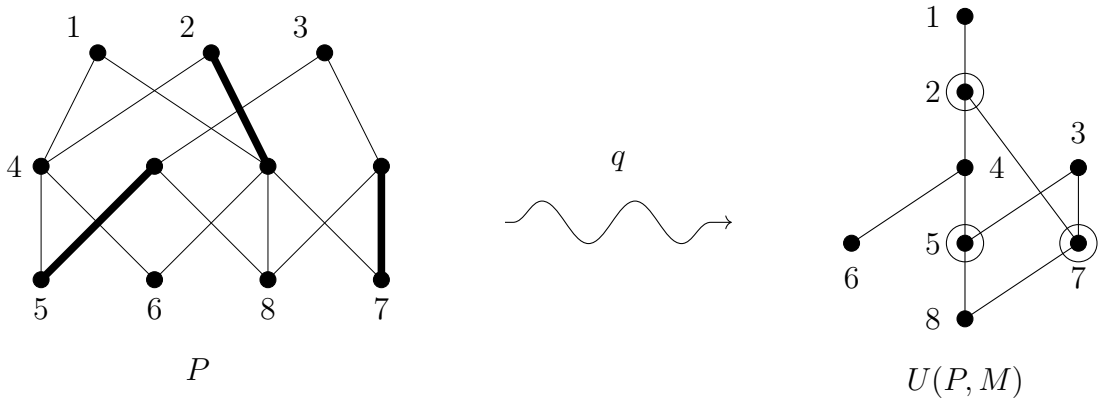


Figure 2.1: *An example of a universal poset. Circles indicate elements with fibers of cardinality 2.*

Theorem 2.10 (Universality of $U(P, M)$)**.**
For any poset P, and for any acyclic matching M on P, we have:

(1) the partial order on $U(P, M)$ is well-defined;

(2) the induced quotient map $q : P \to U(P, M)$ is a poset map with small fibers;

(3) the linear extensions of $U(P, M)$ are in 1-to-1 correspondence with collapsing orders for M.

More precisely, this correspondence is given by the composition of the quotient map with a linear extension.

Proof. To prove the first part we need to check the three axioms of partial orders. Reflexivity and transitivity are obvious, since in Definition 2.9 we have taken the transitive closure. We need to prove only the antisymmetry. Assume that it does not hold, so we have $X, Y \in U(P, M)$ such that $X \leq_U Y, Y \leq_U X$, and $X \neq Y$. Choose a sequence

$$X <_U S_1 <_U \cdots <_U S_p <_U Y <_U T_1 <_U \cdots <_U T_q <_U X, \tag{2.2}$$

with the minimal possible p and q. All the sets $S_1, \ldots, S_p$ and $T_1, \ldots, T_q$ must have cardinality 2, since we choose p and q to be minimal.

Let us discuss first the case $p = q = 0$. If $|X| = |Y| = 1$, say $X = \{x\}$ and $Y = \{y\}$, then we have $x \leq y$ and $y \leq x$, implying $x = y$.
If $|X| = 1$ and $|Y| = 2$, say $X = \{x\}$ and $Y = (a, b)$, then $b > x$ and $x > a$, and this gives $b > x > a$, contradicting the assumption that b covers a. By symmetry of (2.2), this argument includes also the case $|Y| = 1$, $|X| = 2$, so we now assume that $|X| = |Y| = 2$. In this case $X <_U Y <_U X$ is a cycle, contradicting the acyclicity of our matching.

Now, we deal with $p+q \geq 1$. Assume first $|X| = |Y| = 1$, say $X = \{x\}$ and $Y = \{y\}$. If $p = 0$ and $q = 1$, let $T_1 = (a, b)$, then we have $x \leq y, b \geq y$, and $x \geq a$; that leads us to $b \geq y \geq x \geq a$, implying $x = y$, since b covers a. Again by symmetry, we take care of the case $p = 1$ and $q = 0$ as well.
Without loss of generality, we can now assume that $p + q \geq 2$, or $|Y| = 2$ and $p + q \geq 1$. In the first case,

$$S_1 <_U \cdots <_U S_p <_U T_1 <_U \cdots <_U T_q$$

gives a cycle, contradicting the acyclicity of the matching; in the second case such a cycle is given by

$$S_1 <_U \cdots <_U S_p <_U Y <_U T_1 <_U \cdots <_U T_q.$$

For the second part, if $x < y$ in P and $x \in X$, $y \in Y$, for $X, Y \in U(P, M)$, then $X \leq Y$ by Definition 2.9. So q is a poset map, also the fibers are small from the proof of the first part.

Now we prove the last part. Given a linear extension $l : U(P, M) \to Q$, the composition $l \circ q : P \to Q$ is a poset map with small fibers, and it is surjective since both l and

q are.
Conversely, let $\varphi : P \to Q$ be a collapsing order for M. Since φ is surjective, φ^{-1} is nonempty for every $x \in Q$; actually, we have a bijection between sets φ^{-1}, for $x \in Q$, and elements of $U(P, M)$. We set $l(q(\varphi^{-1}(x))) := x$, for each $x \in Q$, thus $l \circ q = \varphi$ as set maps. Now, observe that an elementary relation $S \geq T$, for $S, T \in U(P, M)$, implies that there exist $x \in S$ and $y \in T$ such that $x \geq y$, which implies $\varphi(x) \geq \varphi(y)$, since φ is order-preserving, then l is order-preserving as well; notice that all relations are just the transitive closures of the elementary ones.
Thus, we factor φ through $U(P, M)$, and we get the desired 1-to-1 correspondence. □

2.3 Patchwork Theorem

Viewing the poset maps with small fibers as the central notion of the combinatorial part of discrete Morse theory is very useful for the structural explanation of a standard way to construct acyclic matchings on fibers of a poset map.

Definition 2.11. *A **poset fibration** is a pair $(B, \mathcal{F})$, where*

- *B is a poset, thought of as the **base** of the fibration;*
- *$\mathcal{F} = \{F_x\}_{x \in B}$ is a collection of posets, indexed by the elements of B, thought of as individual **fibers**.*

Associated to such a fibration, we define a poset $E(B, \mathcal{F})$ whose set of elements is $\bigcup_{x \in B} F_x$, and with the order relation given by $\alpha \geq \beta$ if either $\alpha, \beta \in F_x$ and $\alpha \geq \beta$ in F_x, for some $x \in B$, or $\alpha \in F_x$, $\beta \in F_y$, and $x > y$ in B. This poset is called the *total space*.
We also have a poset map $p : E(B, \mathcal{F}) \to B$ defined by $p(\alpha) = x$ if $\alpha \in F_x$. In particular, we have $p^{-1}(x) = F_x$, for all $x \in B$. This is the *projection map* of the total space to the base space, whose preimages are the fibers.
The notion of poset fibration satisfies the following universal property.

Theorem 2.12 (Decomposition theorem)**.**
For an arbitrary poset fibration $(B, \mathcal{F})$, where $\mathcal{F} = \{F_x\}_{x \in B}$, and an arbitrary poset P, there is a 1-to-1 correspondence between:

- *poset maps $\varphi : P \to E(B, \mathcal{F})$;*
- *pairs $(\psi, \{g_x\}_{x \in B})$, where both $\psi : P \to B$ and $g_x : \psi^{-1}(x) \to F_x$ are poset maps, for each $x \in B$.*

Under this bijection, the fibers of φ are the same as the fibers of the maps g_x.

Proof. Let us define a bijection. One direction is trivial: given a poset map $\varphi : P \to E(B, \mathcal{F})$, composing it with the projection map $p : E(B, \mathcal{F}) \to B$ gives us the poset map $\psi : P \to B$, then we obtain the poset maps g_x by taking the appropriate restrictions of φ.

In the opposite direction, assume that we have a poset map $\psi : P \to B$ and a collection of poset maps $g_x : \psi^{-1}(x) \to F_x$, for all $x \in B$. Now define $\varphi : P \to E(B, \mathcal{F})$ by taking the value of the reasonable fiber map:

$$\varphi(\alpha) := g_{\psi(\alpha)}(\alpha),$$

for all $\alpha \in P$. If we have $\alpha, \beta \in P$ such that $\alpha > \beta$, then $\psi(\alpha) \geq \psi(\beta)$, since ψ is a poset map. Now if $\psi(\alpha) = \psi(\beta)$, we have $g_{\psi(\alpha)}(\alpha) \geq g_{\psi(\alpha)}(\beta) = g_{\psi(\beta)}(\beta)$, since $g_{\psi(\alpha)}$ is a poset map. Otherwise, we have $\psi(\alpha) > \psi(\beta)$, so $g_{\psi(\alpha)}(\alpha) > g_{\psi(\beta)}(\beta)$ by the definition of the partial order on the total space. □

The decomposition theorem is often used as a rationale to construct an acyclic matching on a poset P: first map P to some other poset Q, then construct acyclic matchings on the fibers of this map; these matchings will "patch together" to form an acyclic matching for P.

Theorem 2.13 (Patchwork theorem)**.**
Let $\varphi : P \to Q$ be a poset map, and assume that we have acyclic matchings on subposets $\varphi^{-1}(q)$, for all $q \in Q$. Then the union of these matchings is an acyclic matching on P.

Proof. Consider the poset Q as the base space, and the fibers maps g_q given by the acyclic matchings on the subposets $\varphi^{-1}(q)$.
The decomposition theorem states that there exists a poset map from P to the total space of such a fibration, and that the fibers of this map are the same as the fibers of the maps g_q. Since the second are given by acyclic matchings, we have a poset map from P with small fibers that corresponds exactly to the union of acyclic matchings on the subposets $\varphi^{-1}(q)$, for $q \in Q$. □

2.4 Three classical results

Internal collapses on the boundary of a simplex

Let Δ be the boundary of an n-dimensional simplex. We see that $\mathcal{F}(\Delta) \setminus \{\hat{0}\} = \bar{\mathcal{B}}_{n+1}$; see Example 1.18. Consider the following matching M on $\bar{\mathcal{B}}_{n+1}$:

$$(S, S \cup \{1\}) \in M \text{ for all } S \subseteq \{2, \ldots, n+1\}.$$

Clearly, in this way we obtain an acyclic matching; in particular, the only critical simplices are $\{1\}$ and $\{2, \dots, n+1\}$. It follows that $\Delta \simeq \mathbb{S}^{n-1}$.

Independence complexes of strings and cycles

For an arbitrary integer $n \geq 1$, we denote with L_n the graph consisting of n vertices and $n-1$ edges that connect these vertices forming a string.

Proposition 2.14. *For any $n \geq 1$, we have*

$$\mathrm{Ind}(L_n) \simeq \begin{cases} \mathbb{S}^{k-1}, & \text{if } n = 3k; \\ \mathrm{pt}, & \text{if } n = 3k+1; \\ \mathbb{S}^{k}, & \text{if } n = 3k+2. \end{cases}$$

Proof. We can assume that the vertices of L_n are labeled 1 through n in the same sequence as they occur along the string. Let k denote the maximal integer such that $3k \leq n$. Furthermore, let C be a chain with $k+1$ elements labeled as follows:

$$c_3 > c_6 > \cdots > c_{3k} > c_r.$$

We define a map $\varphi : \mathcal{F}(\mathrm{Ind}(L_n)) \to C$ in the following way. The simplices that contain the vertex labeled 3 get mapped to c_3; the simplices that do not contain the vertex labeled 3, but contain the one labeled 6 get mapped to c_6; the simplices that do not contain the vertices labeled 3 and 6, but contain the one labeled 9, get mapped to c_9; and so on. Finally, the simplices that do not contain any of the vertices labeled $3, \dots, 3k$, all get mapped to c_r (r stands for "the rest").
Clearly, this map is order-preserving, since if one takes a larger simplex, it will have more vertices, and so its image may only go up in the chain.

Let us now define acyclic matchings on the fibers of C under φ. We split our argument into three cases.
Case 1. First we consider the fiber $\varphi^{-1}(c_3)$. For any simplex $\sigma \in \varphi^{-1}(c_3)$ we have $3 \in \sigma$, thus $2 \notin \sigma$. Therefore pairing σ with $\sigma \cup \{1\}$ provides a matching that is well-defined and acyclic.
Case 2. Next, we consider $\varphi^{-1}(c_6), \dots, \varphi^{-1}(c_{3k})$. Let t be an integer such that $2 \leq t \leq k$. The fiber $\varphi^{-1}(c_{3t})$ consists if all simplices σ such that $3, 6, \dots, 3t-3 \notin \sigma$, while $3t \in \sigma$. Since $3t-1 \notin \sigma$, the pairing $(\sigma, \sigma \cup \{3t-2\})$ provides a well-defined acyclic matching.
Case 3. Finally, we consider the fiber $\varphi^{-1}(c_r)$. Now we have three subcases.
If $n = 3k+1$, then this fiber is a face poset with a cone with apex in n; in particular, the pairing $(\sigma, \sigma \cup \{n\})$ provides an acyclic matching with only one critical cell $\{n\}$.

Therefore by Theorem 2.13 $\mathrm{Ind}(L_{3k+1})$ is collapsible.
If $n = 3k$, we see that $\varphi^{-1}(c_r)$ is a face poset of the boundary of a k-dimensional cross-polytope[1], which is homeomorphic to $\mathbb{S}^{k-1}$. Again by Theorem 2.13 we have that $\mathrm{Ind}(L_{3k})$ is homotopy equivalent to $\mathbb{S}^{k-1}$.
If $n = 3k+2$, we see that $\varphi^{-1}(c_r)$ is a face poset of the boundary of a $(k+1)$-dimensional cross-polytope, which is homeomorphic to $\mathbb{S}^k$. The rest is the same, and we conclude that $\mathrm{Ind}(L_{3k+2})$ is homotopy equivalent to $\mathbb{S}^k$. □

Note that the above proof actually yields a stronger statement: in fact we get a collapsibility instead of just contractibility. In particular, we get a sequence of collapses leading to a precise sphere, sitting inside $\mathrm{Ind}(L_n)$ as a subcomplex.

Now, for an arbitrary integer $n \geq 2$, we let C_n denote the cycle with n vertices labeled $1, \ldots, n$. The homotopy type of the independence complexes of cycles can be easily described as well.

Proposition 2.15. *For any $n \geq 2$, we have*

$$\mathrm{Ind}(C_n) \simeq \begin{cases} \mathbb{S}^{k-1} \vee \mathbb{S}^{k-1}, & \text{if } n = 3k; \\ \mathbb{S}^{k-1}, & \text{if } n = 3k \pm 1. \end{cases}$$

Proof. Let k denote the maximal integer such that $3k \leq n+1$, and let the chain C be defined on the same way as in Proposition 2.14. Let also $\varphi : \mathcal{F}(\mathrm{Ind}(C_n)) \to C$ be the order-preserving map described by the same rule as the one in Proposition 2.14. Now we look for acyclic matchings on the fibers.

First, the matchings on the fibers $\varphi^{-1}(c_6)$ through $\varphi^{-1}(c_{3k})$ are in the same way as in Proposition 2.14, and they are again well-defined and acyclic, without critical cells. The remaining two cases are a bit different.
The fiber $\varphi^{-1}(c_3)$ is the same as the face poset of $\mathrm{Ind}(L_{n-3})$ with an added minimal element. Thus taking the acyclic matching for $\mathrm{Ind}(L_{n-3})$ and extend it by matching the critical 0-cell with the minimal element yields a new acyclic matching. Now if $n = 3k+1$, this matching has no critical cells at all. Otherwise, if $n = 3k$ or $n = 3k-1$, it has one critical cell of dimension $k-1$.

Finally, we describe an acyclic matching on $\varphi^{-1}(c_r)$ by considering three cases.
If $n = 3k-1$, we know that $3, 6, \ldots, 3k \notin \sigma$; where with our conventions $3k = 1$. Therefore, we have a face poset of a cone with apex in 2, hence the pairing $(\sigma, \sigma \cup \{2\})$ gives a well-defined acyclic matching with one critical cell $\{2\}$.
If $n = 3k$, then we have a face poset of the join of k copies of $\mathbb{S}^0$. Denote the sets of

[1] See Example 1.18.

vertices of these k copies by $\{x_1, y_1\}, \ldots, \{x_k, y_k\}$, and consider the pairing $(\sigma, \sigma \cup \{x_i\})$, where i is the minimal index such that $y_i \notin \sigma$. This is a well-defined acyclic matching with critical cells $\{x_1\}$ and $\{y_1, \ldots, y_k\}$.
If $n = 3k + 1$, then we have a face poset of $k - 1$ copies of $\mathbb{S}^0$ and one of $\mathrm{Ind}(L_3)$. Denote the sets of vertices of these $k - 1$ copies by $\{x_1, y_1\}, \ldots, \{x_{k-1}, y_{k-1}\}$, and let $\{x_k, y_k, z_k\}$ be the vertices of $\mathrm{Ind}(L_3)$, with y_k being the middle vertex. Consider now the same pairing: $(\sigma, \sigma \cup \{x_i\})$, where i is the minimal index such that $y_i \notin \sigma$. We obtain a well-defined acyclic matching with critical cells $\{x_1\}$ and $\{y_1, \ldots, y_k\}$. □

The face poset of the partition lattice

Recall the partition lattice Π_n introduced in Preliminaries.

Theorem 2.16. *For $n \geq 3$, the simplicial complex $\Delta(\bar{\Pi}_n)$ is homotopy equivalent to a wedge of $(n-1)!$ spheres of dimension $n-3$.*

Proof. The statement is clearly true for $n = 3$, so from now on we assume $n \geq 4$ and proceed by induction. Set $\alpha := (1)(2, 3, \ldots, n)$, and let Q to be the interval consisting of all partitions having a singleton block (1) with reversed order, that formally is , $Q := [\hat{0}, \alpha]^{op}$. We define an order-preserving map $\varphi : \mathcal{F}(\Delta(\bar{\Pi}_n)) \to Q$ by the following rule:

$$\mathcal{F}(\Delta(\bar{\Pi}_n)) \ni c \mapsto q \in Q, \text{ with } q \text{ being the minimal element that can be added to } c.$$

Let us analyze this rule; take $c \in \mathcal{F}(\Delta(\bar{\Pi}_n))$, assume $c = (\pi_1 < \pi_2 < \cdots < \pi_t)$, and consider two cases.
Case 1. If $\alpha \geq \pi_t$, then $\varphi(c) = \alpha$.
Case 2. If $\alpha \not\geq \pi_k$ and either $\alpha \geq \pi_{k-1}$ or $k = 1$, then $\varphi(c) = \pi_k \wedge \alpha$.[2]
In other words, find the smallest partition π_k in c where 1 is a part of a non-singleton block B, and then partition B into (1) and $B \setminus \{1\}$. This explanation also shows that the minimal element in this rule is unique, thus the map is well-defined.

This map is order-preserving, since we note that if the chain is increased, then the minimal possible element of Q, that is, the maximal possible element of $[\hat{0}, \alpha]$ that can be added to this chain will either remain the same or increase in Q.

By Theorem 2.13 it is now sufficient to construct acyclic matchings on the fibers of φ. We have two cases.
Case 1. Let $S = \varphi^{-1}((1)(2)\ldots(n))$. Clearly, the poset S is actually a disjoint union

[2]The wedge $\wedge$ here represents the lattice operation; see the Lattices subsection in Chapter 1.

$S = S_2 \cup \cdots \cup S_n$, where S_i is the subposet consisting of all chains containing the element $(1i)(2)\dots(i-1)(i+1)\dots(n)$, for $i = 2, \dots, n$. Furthermore, each S_i is actually a copy of $\mathcal{F}(\Delta(\bar{\Pi}_{n-1})) \cup \{\hat{0}\}$. By induction, there exists an acyclic matching on $\mathcal{F}(\Delta(\bar{\Pi}_{n-1}))$ that has one critical cell in dimension 0 and $(n-2)!$ critical cells in dimension $n-4$. Now, in $\mathcal{F}(\Delta(\bar{\Pi}_{n-1})) \cup \{\hat{0}\}$ this matching can be extended to have only the top-dimensional critical elements, since we can match with $\hat{0}$ the remaining one. When considered in S_i, these maximal chains consist of $n-2$ elements; therefore they corresponds to critical simplices of dimension $n-3$ in $\Delta(\bar{\Pi}_n)$.

Case 2. Let $S = \varphi^{-1}(\pi)$, for $\pi \neq (1)(2)\dots(n)$. The matching rule in this case is to add π to the chain if it is not there already; otherwise, remove it. Clearly this gives an acyclic matching. The only critical element is the chain consisting of only π: This corresponds to one critical cell of dimension 0.

To summarize, we get $(n-1)(n-2)! = (n-1)!$ critical cells of dimension $n-3$ and one critical cell of dimension 0. Therefore we may conclude that $\Delta(\bar{\Pi}_n)$ is homotopy equivalent to a wedge of $(n-1)!$ spheres of dimension $n-3$. These spheres are enumerated by the critical cells of dimension $n-3$. □

Chapter 3

NP-completeness

The **running time** of an algorithm on a particular input is the number of instructions and data access executed. Clearly, how we compute these costs should be independent of any particular computer. We agree that executing each line of pseudocode requires a constant amount of time; this viewpoint reflects how the pseudocode would actually be implemented.

Actually, we make one more simplifying abstraction: we are interested in the **order of growth** of the running time.

3.1 Problems and complexity

Polynomial-time solvability

An **abstract problem** Q is a binary relation on a set I of problem **instances** and a set S of problem **solutions**. Q is called a **decision problem** if it is a function and $S = \{0, 1\}$.
The theory of NP-completeness restricts attention to decision problems and, in this case, we can view an abstract decision problem as a function mapping I to $\{0, 1\}$.

An **encoding** of a set S of abstract objects is a mapping $e : S \to \{0, 1\}^*$, where $\{0, 1\}^*$ is the set of binary strings. A computer algorithm actually takes an encoding of an instance as input. The **size** of an instance i is the length of its string $e(i)$ and we denote it by $|i|$.

A **concrete problem** is a problem whose instances are binary strings. We say that an algorithm **solves** a concrete problem in $O(T(n))$ time if, when its input is a problem instance of length $n = |i|$, the algorithm's output is produced in $O(T(n))$ time.
A concrete problem is **polynomial-time solvable** if there exists an algorithm that

solves it in $O(n^k)$ time for some constant k. The complexity class **P** is the set of concrete decision problems that are polynomial-time solvable.

Encodings map abstract problems to concrete problems and so we would like to extend the definition of polynomial-time solvability to abstract problems. Unfortunately, it depends on the encoding but, in practice, the actual encoding of a problem makes little difference to the polynomial-time solvability.
We say that a function $f : \{0,1\}^* \to \{0,1\}^*$ is **polynomial-time computable** if there exists a polynomial-time algorithm A that, given any input $x \in \{0,1\}^*$, produces as output $f(x)$. We say that two encodings e_1 and e_2 of a set I are **polynomially related** if there exist two polynomial-time computable functions f_{12} and f_{21} such that, for any $i \in I$, we have $f_{12}(e_1(i)) = e_2(i)$ and $f_{21}(e_2(i)) = e_1(i)$.

Lemma 3.1. *Let Q be an abstract decision problem on an instance set I, and let e_1 and e_2 be polynomially related encodings on I. Then, $e_1(Q) \in P$ if and only if $e_2(Q) \in P$.*

Proof. We prove the forward direction, then the backward one is symmetric.
Suppose that $e_1(Q)$ can be solved in $O(n^k)$ time for some constant k. Suppose also that for any problem instance i, we can compute the encoding $e_1(i)$ from $e_2(i)$ in $O(n^c)$ time for some constant c, where $n = |e_2(i)|$. Solving problem $e_2(Q)$ on input $e_2(i)$ requires the computation of $e_1(i)$, and then we run the algorithm for $e_1(Q)$ on $e_1(i)$. Converting encodings takes $O(n^c)$ time, and therefore $|e_1(i)| = O(n^c)$, since the output of a serial computer cannot be longer than its running time. Then, the problem on $e_1(i)$ is solved in $O(|e_1(i)|^k) = O(n^{ck})$ time, which is polynomial. □

A formal-language framework

An **alphabet** Σ is a finite set of symbols. We denote the set of all strings over Σ by Σ^*. A **language** L over Σ is a subset of Σ^*. We denote the *empty string* by ε and the *empty language* by $\emptyset$.

From the point of view of language theory, the set of instances for any decision problem Q is simply Σ^*, where the alphabet Σ is $\{0,1\}$. We can view Q as a language $L = \{x \in \{0,1\}^* : Q(x) = 1\}$.
We say that an algorithm A **accepts** a string $x \in \{0,1\}^*$ if the algorithm terminates on x and its output $A(x)$ is 1; if the algorithm terminates on x and $A(x) = 0$ we say that A **rejects** x.
A language L is **decided** by an algorithm A if every string in L is accepted by A and every string not in L is rejected by A. L is decided in polynomial time if it is decided

and if exists a constant k such that, for any length-n string x, the algorithm correctly decides whether $x \in L$ in $O(n^k)$ time.

Using this framework, we can provide an alternative definition of P:

$$\mathrm{P} = \{L \subseteq \{0,1\}^* : L \text{ is decided in polynomial time by some algorithm } A\}.$$

In fact P is also the class of languages that can be accepted in polynomial time.

Theorem 3.2. *$P = \{L : L$ is accepted in polynomial time by an algorithm $\}$.*

Proof. The class of languages decided in polynomial time is a subset of the class of languages accepted in polynomial time, then we need only to show that if L is accepted by a polynomial-time algorithm, we also have a polynomial-time algorithm that decides L.

Let A be an algorithm that accepts L in $O(n^k)$ time for some constant k. Then we have another constant c such that A accepts L in at most cn^k steps. Now let A' be an algorithm that, given any input string x, simulates cn^k steps of A. After these steps, A' inspects the behavior of A.
If A has accepted x, then A' accepts x by giving a 1 in output. Otherwise, if A has not accepted x, then A' rejects x by giving a 0 in output. Running A' increase the running time of A only by a polynomial factor, ans thus A' is a polynomial-time algorithm that decides L. □

Polynomial-time verification

We define a **verification algorithm** as being a two-argument algorithm A, where one argument is the input string x and the other is a string y called a **certificate**. Such an algorithm **verifies** an input string x if there exists a certificate y such that $A(x, y) = 1$.

The complexity class **NP** is the class of languages that can be verified by a polynomial-time algorithm. Formally, a language L belongs to NP if and only if there exists a two-input polynomial-time algorithm A and a constant c such that

$$L = \{x \in \{0,1\}^* : \exists\, y \text{ with } |y| = O(x^c) \text{ such that } A(x, y) = 1\}.$$

We can see that P $\subseteq$ NP, since if there is a polynomial-time algorithm to decide L, it can be converted in a two-argument algorithm that simply ignores any certificate.
A definitive answer for whether P = NP is unknown.

3.2 Reducibility

One way that sometimes works for solving a problem is to recast it as a different one. If a problem Q reduces to another problem Q', then Q is "no harder to solve" than Q'. We say that a language L_1 is **polynomial-time reducible** to a language L_2, written $L_1 \leq_{\mathrm{P}} L_2$, if there exists a polynomial-time computable function $f : \{0,1\}^* \to \{0,1\}^*$ such that for all $x \in \{0,1\}^*$

$$x \in L_1 \text{ if and only if } f(x) \in L_2.$$

The function f is the *reduction function*, and a polynomial-time algorithm that computes it is a *reduction algorithm*.

Lemma 3.3. *If $L_1 \leq_P L_2$, then $L_2 \in P$ implies $L_1 \in P$.*

Proof. Let A_2 be a polynomial-time algorithm that decides L_2, and let F be a polynomial-time reduction algorithm that computes the reduction function f, which reduces L_1 to L_2.
We now construct a polynomial-time algorithm A_1 that decides L_1. For a given input $x \in \{0,1\}^*$, A_1 uses F to transform x into $f(x)$, and then it uses A_2 to establish whether $f(x) \in L_2$. The output of A_1 is that of A_2. □

NP-completeness

We can now define the set of NP-complete languages.
A language L is **NP-complete** if:

(i) $L \in \mathrm{NP}$;

(ii) $L' \leq_{\mathrm{P}} L$ for every $L' \in \mathrm{NP}$.

If a language satisfies property 2 is said to be **NP-hard**. **NPC** is the class of NP-complete languages. This class has the following important property.

Theorem 3.4. *If any NP-complete problem is polynomial-time solvable, then $P = \mathrm{NP}$.*

Proof. Suppose that exists $L \in \mathrm{P} \cap \mathrm{NPC}$. For any $L' \in \mathrm{NP}$, we have $L' \leq_{\mathrm{P}} L$ by definition of NP-completeness. Thus, by Lemma 3.3, we also have $L' \in \mathrm{P}$, which proves the statement. □

The following lemma provides a foundation for proving that a given language is NP-complete.

Lemma 3.5. *If exists* $L' \in \text{NPC}$ *such that* $L' \leq_P L$*, then* L *is NP-hard. If, in addition, we have* $L \in \text{NP}$*, then* L *is NP-complete.*

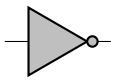
Proof. Since L' is NP-complete, for all $L'' \in \text{NP}$, we have $L'' \leq_\text{P} L'$. We also have $L' \leq_\text{P} L$ by supposition, thus by transitivity we prove $L'' \leq_\text{P} L$, which is the NP-hardness of L. Now, proving the second statement is trivial. □

3.3 Some NP-complete problems

We have discussed the notion of NP-completeness, but until now, we have not proved that any problem is NP-complete. Once we prove that at least one problem is NP-complete, polynomial-time reductions become tools for proving that other problems are NP-complete.

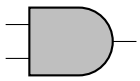
CIRCUIT-SAT

A **boolean combinational element** is any circuit element that has a constant number of boolean inputs and outputs, and that performs a well-defined function. For the circuit-satisfiability problem we need only **logic gates**.



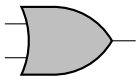
x	$\neg x$
0	1
1	0

(a) *NOT*



x	y	$x \wedge y$
0	0	0
0	1	0
1	0	0
1	1	1

(b) *AND*



x	y	$x \vee y$
0	0	0
0	1	1
1	0	1
1	1	1

(c) *OR*

Figure 3.1: *Three basic logic gates and their truth tables.*

A **boolean combinational circuit** consists of one or more boolean combinational elements interconnected by wires so that there are no cycles. The number of element inputs fed by a wire is called the *fan-out* of the wire. If no element output is connected to a wire, the wire is a *circuit input*. Likewise, if no element input is connected to a wire, it is called a *circuit output*. For the purpose of defining the circuit-satisfiability problem, we limit the number of circuit outputs to 1.

A **truth assignment** is a set of boolean input values. We say that boolean combinational circuit is **satisfiable** if there exists a *satisfying assignment*, that is, a truth assignment that raise a 1 in the circuit output.

The **circuit-satisfiability** problem is: "Given a boolean combinational circuit made up by AND, OR, and NOT gates, is it satisfiable?". In order to formalize this question, we must introduce a standard encoding for circuits.

The **size** of a boolean combinational circuit is the number of boolean combinational elements plus the number of wires in the circuit. We agree on a graph-like encoding that maps any given circuit C into a binary string $\langle C \rangle$ whose length is polynomial in the size of C. We can now define

$$\text{CIRCUIT-SAT} = \{\langle C \rangle : C \text{ is a satisfiable boolean combinational circuit}\}$$

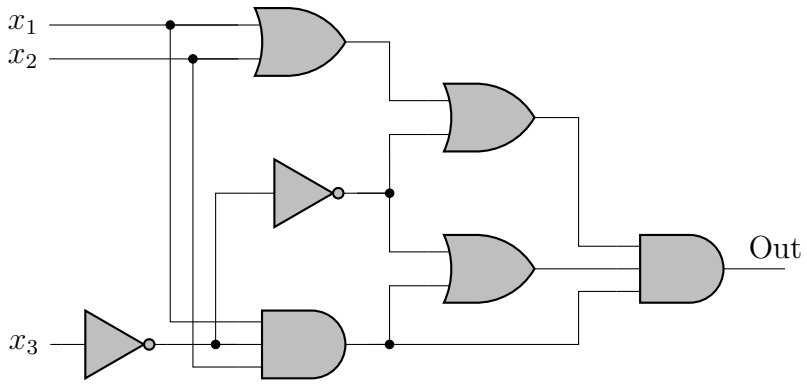


Figure 3.2: *An example of a boolean combinational circuit.*

Example 3.6. Observe that the circuit in fig. 3.2 is satisfiable, since the input $\langle x_1 = 1, x_2 = 1, x_3 = 0 \rangle$ sets the output to 1.
On the other hand, if we substitute one of the two OR gates on the right with an AND gate, then the circuit is no longer satisfiable.

Given a circuit, one can determine whether is satisfiable by simply checking all its possible input assignments. Unfortunately, if the circuit has k inputs, this strategy requires to check up to 2^k possible assignments.

Lemma 3.7. *CIRCUIT-SAT belongs to NP.*

Proof. We provide a two-input, polynomial-time algorithm A that verifies CIRCUIT-SAT. One of the inputs to A is a boolean combinational circuit C—actually a standard encoding of it. The other input is a certificate corresponding to an assignment of a

boolean value to each of the wires in C.
The algorithm A works as follows. For each logic gate in C, it checks that the value provided by the certificate on the output wire is consistent with the actual wire output, which is obtained through the computation of inputs by the logic gate. Then, if the output of the entire circuit is 1, algorithm A outputs 1. Otherwise, the output of A is 0.

Whenever a satisfiable circuit C is given as input to A, there exists a certificate whose length is polynomial in the size of C that causes A to output a 1. On the other hand, if A has an unsatisfiable circuit as input, no certificate can mislead A into outputting 1. A also runs in polynomial time. □

Now we have to show that the language is NP-hard.
A computer program is stored in the computer's memory as a sequence of instructions. A typical instruction encodes an operation to be performed, addresses of operands in memory, and an address where the result should be stored. A special memory location, called the *program counter*, keeps track of which instruction is to be executed next. We call any particular state of the computer memory a *configuration*.
When an instruction executes, it transforms the configuration, and so we think of an instruction as mapping one configuration to another. The computer hardware that realizes this mapping can be implemented as a boolean combinational circuit.

Lemma 3.8. *CIRCUIT-SAT is NP-hard.*

The proof requires some computer science technicalities and we omit it for a better readability. It can be found in Appendix.

Now the next statement follows by definition thanks to these two lemmas.

Theorem 3.9. *CIRCUIT-SAT is NP-complete.*

SAT

This problem has the historical honor of being the first problem ever shown to be NP-complete.
We formulate the **formula satisfiability** problem in terms of the language **SAT**. An instance of SAT is a boolean formula ϕ composed of:

(i) n boolean variables: $x_1, \ldots, x_n$;

(ii) m boolean connectives, i.e. any boolean function with one or two inputs, and one output;

(iii) parentheses.

Follows an example of such a formula:

$$\phi = ((x_1 \to x_2) \vee \neg((\neg x_1 \leftrightarrow x_3) \vee x_4)) \wedge \neg x_2.$$

Without loss of generality, we assume that there are no redundant parentheses.
A boolean formula can be encoded in a string with a length that is polynomial in $n+m$. So we have:

$$\text{SAT} = \{\langle\phi\rangle : \phi \text{ is a satisfiable boolean formula}\}$$

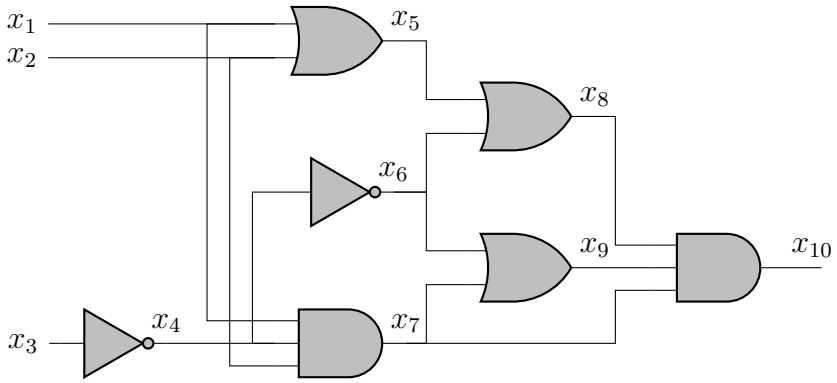
Theorem 3.10. *Satisfiability of boolean formulas is NP-complete.*

Proof. A certificate consists of a satisfying assignment; the verifying algorithm simply replaces each variable in the formula with its corresponding value and then evaluates the expression. Now we show that CIRCUIT-SAT $\leq_{\text{P}}$ SAT.
To express any boolean combinational circuit as a boolean formula we can simply look at the gate that produces the circuit output and inductively express each of the gate's input as formulas. Unfortunately, this method does not amount to a polynomial-time reduction.
To overcome this problem, for each wire x_i in the circuit C, the formula ϕ has a variable x_i. To express how each gate operates, construct a small formula using the variable of its incident wires. The formula has the form of an "if and only if", with the variable for the gate's output on the left and on the right a logical expression representing the gate's function on its inputs. Each of these small formulas is called a *clause*.
The formula ϕ describing C produced by the reduction algorithm is the AND of the circuit output variable with the conjunction of gate clauses.
If C has a satisfying assignment, then each wire has a well-defined value, and the output of C is 1. Therefore, when we assign values to variables in ϕ, each clause evaluates to 1, and thus the conjunction does as well. On the other hand, if some assignment causes ϕ to evaluate to 1, the circuit C is satisfiable by an analogous argument. □

Example 3.11. If we consider the circuit in fig. 3.2, executing the reduction algorithm described above leads us to the following formula:

$$\begin{aligned}\phi = x_{10} &\wedge (x_4 \leftrightarrow \neg x_3) \\ &\wedge (x_5 \leftrightarrow (x_1 \vee x_2)) \\ &\wedge (x_6 \leftrightarrow \neg x_4) \\ &\wedge (x_7 \leftrightarrow (x_1 \wedge x_2 \wedge x_4)) \\ &\wedge (x_8 \leftrightarrow (x_5 \vee x_6)) \\ &\wedge (x_9 \leftrightarrow (x_6 \vee x_7)) \\ &\wedge (x_{10} \leftrightarrow (x_7 \wedge x_8 \wedge x_9)).\end{aligned}$$

Where each x_i represents a gate's output as shown below.



3-CNF-SAT

Reducing from SAT is a more convenient way to prove NP-completeness. However, the reduction algorithm must handle any input formula, and this can lead to a huge number of cases to consider. It is usually simpler to reduce from a restricted language of boolean formulas, that is **3-CNF-SAT**.

A *literal* in a boolean formula is an occurrence of a variable or its negation. A *clause* is the OR of two or more literals. A boolean formula is in **3-conjunctive normal form**, or 3-CNF, if it is expressed as an AND of clauses and each of them contains exactly three distinct literals.
The language 3-CNF-SAT consists of encodings of boolean formulas in 3-CNF that are satisfiable.

Theorem 3.12. *Satisfiability of 3-CNF boolean formulas is NP-complete.*

Proof. The argument from the proof of Theorem 3.10 to show SAT $\in$ NP applies equally well here. Therefore, by Lemma 3.5, we only need to reduce in polynomial time SAT to 3-CNF-SAT.

We break the reduction algorithm in three basic steps.
First, construct a binary tree for ϕ, with literals as leaves and connectives as internal nodes. If the input formula contains a clause such as the OR of several literals, using associativity we can parenthesize the expression so that every internal node in the resulting tree has just one or two children.
Now we introduce a variable y_i for the output of each internal node, like in the proof of Theorem 3.10. So the original formula can be expressed as the AND of the variable at the root of the tree and a conjunction of clauses describing the operations of each node.

The formula ϕ' obtained is a conjunction of clauses ϕ'_i, each of which has at most three literals. However, these clauses are not yet ORs of three literals.

The second step of the reduction converts each clause ϕ'_i in CNF. We construct a truth table for ϕ'_i by evaluating all possible assignments to its variables; now, using the assignments that evaluate to 0, we build a formula in *disjunctive normal form*—an OR of ANDs—that is equivalent to $\neg\phi'_i$. Then negate this formula and use *De Morgan's laws* lead us to a CNF formula ϕ''_i.
At this point, each clause of ϕ' has been converted into a CNF formula ϕ''_i, and thus ϕ' is equivalent to the CNF formula ϕ'' consisting of the conjunction of the ϕ''_i. Also, each clause of ϕ'' has at most three literals.

The final step transforms the formula so that each clause has exactly three distinct literals. From ϕ'', we construct the 3-CNF formula ϕ''' that uses two auxiliary variables, p and q. For each clause C_i of ϕ'', we distinguish three cases:

- If C_i contains three distinct literals, then we simply include C_i as a clause of ϕ'''.
- If C_i contains exactly two distinct literals, that is, if $C_i = (l_1 \vee l_2)$, then include $(l_1 \vee l_2 \vee p) \wedge (l_1 \vee l_2 \vee \neg p)$ as clauses of ϕ'''.
- If C_i contains just one distinct literal l, then include $(l \vee p \vee q) \wedge (l \vee p \vee \neg q) \wedge (l \vee \neg p \vee q) \wedge (l \vee \neg p \vee \neg q)$ as clauses of ϕ'''.

We can see that ϕ''' is satisfiable if and only if ϕ is satisfiable. We have already seen that the construction of ϕ' from ϕ preserves satisfiability. The second step produces ϕ'', which is algebraically equivalent to ϕ'. Then the third step produces ϕ''' that is effectively equivalent to ϕ'', since any assignment to the variables p and q produces a formula that is algebraically equivalent to ϕ''.

To conclude we need to show that the reduction can be computed in polynomial time. Constructing ϕ' from ϕ introduces at most one variable and one clause per connective in ϕ. Then ϕ'' is obtained from ϕ' introducing at most eight clauses for each clause from ϕ', since each clause contains at most three variables, and so its truth table has at most eight rows. In the end the construction of ϕ''' from ϕ'' introduces at most four clauses for each on in ϕ''. □

Example 3.13. Here is an example of how the reduction algorithm constructs ϕ''' from

$$\phi = ((x_1 \to x_2) \vee \neg((\neg x_1 \leftrightarrow x_3) \vee x_4)) \wedge \neg x_2.$$

We first construct

$$\begin{aligned}\phi' = y_1 \wedge (y_1 \leftrightarrow (y_2 \wedge \neg x_2)) \\ \wedge (y_2 \leftrightarrow (y_3 \vee y_4)) \\ \wedge (y_3 \leftrightarrow (x_1 \rightarrow x_2)) \\ \wedge (y_4 \leftrightarrow \neg y_5) \\ \wedge (y_5 \leftrightarrow (y_6 \vee x_4)) \\ \wedge (y_6 \leftrightarrow (\neg x_1 \leftrightarrow x_3)),\end{aligned}$$

then for each clause we proceed with step two.
We show an example with the clause $\phi'_1 = (y_1 \leftrightarrow (y_2 \wedge \neg x_2))$ that describes its truth table, the corresponding DNF formula, and the equivalent CNF formula.

y_1	y_2	x_2	ϕ'_1
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

$$\begin{aligned}\neg\phi'_1 =&(y_1 \wedge y_2 \wedge x_2) \vee (y_1 \wedge \neg y_2 \wedge x_2) \\ &\vee (y_1 \wedge \neg y_2 \wedge \neg x_2) \vee (\neg y_1 \wedge y_2 \wedge \neg x_2),\end{aligned}$$

$$\begin{aligned}\neg\phi''_1 =&(\neg y_1 \vee \neg y_2 \vee \neg x_2) \wedge (\neg y_1 \vee y_2 \vee \neg x_2) \\ &\wedge (\neg y_1 \vee y_2 \vee x_2) \wedge (y_1 \vee \neg y_2 \vee x_2).\end{aligned}$$

In this specific case the third step is unnecessary.

Chapter 4

Recognition of Collapsibility

A classical question often considered in algebraic topology is whether some topological space is contractible. We can consider this question as an algorithmic question, that is, we consider the topological space as an input for an algorithm (say as a finite simplicial complex[1]). It turns out that this question is undecidable by a result of Novikov [3]; we briefly discuss this in Appendix.
An important, algorithmically recognizable, subclass of contractible complexes is the class of collapsible complexes. We focus on the computational complexity of the collapsibility problem. First we talk about previous results and we show that this question is NP-complete even if we restrict the input to 3-dimensional complexes; further we will show some corollaries, and we will finish asking the question in the remaining cases. The first main result, proved by Tancer [7], is the following.

Theorem 4.1. *It is NP-complete to decide whether a given 3-dimensional simplicial complex is collapsible.*

Clearly, this problem belongs to NP (it is just sufficient to guess a right sequence of elementary collapses), thus the nontrivial part is showing its NP-hardness. Moreover, by attaching a d-simplex to the complexes used in the proof of Theorem 4.1 it is easy to observe that the statement is also valid if we replace "3-dimensional" with "d-dimensional" for any $d \geq 4$. More details can be founded in the conclusion; see Section 4.5.

Previously, Eğecioğlu and Gonzalez [4] showed that it is NP-complete to decide whether a given 2-dimensional complex can be collapsed to a point by removing at most k triangles, where k is a part of the input. As pointed out by Joswig and Pfetsch [5] and by Malgouyres and Francés [6], this problem becomes polynomial-time solvable

[1]Many topological spaces cannot be represented as finite simplicial complexes. However, we only consider those that can be represented in this way.

when k is fixed. In particular, deciding whether a 2-dimensional complex collapses to a point is polynomial-time solvable. The same approach yields the fact that deciding whether a d-complex collapses to a $(d-1)$-complex is polynomial-time solvable; we will prove this fact later in the dedicated subsection.
Furthermore, Malgouyres and Francés have shown that it is NP-complete to decide whether a given 3-dimensional complex collapses to some 1-dimensional one, and the approach in the proof of Theorem 4.1 relies, in a significant part, on this work.

Using the result of Eğecioğlu and Gonzalez, Joswig and Pfetsch also proved that it is NP-complete to decide whether there exists a Morse matching with at most c critical cells, where c is a part of the input.
Theorem 4.1 can be reformulated in terms of Morse matchings in the following way.

Theorem 4.2. *It is NP-complete to decide whether a given 3-dimensional simplicial complex admits a perfect Morse matching*[2]*.*

Proof. Thanks to the results obtained in Chapter 2, we can use the fact that a simplicial complex K collapses to a point x if and only if it admits a Morse matching such that $K \setminus \{x, \emptyset\}$ contains no critical cells. Therefore, K is collapsible if and only if it admits a perfect Morse matching. Consequently, Theorem 4.2 is equivalent to Theorem 4.1. □

In this chapter we work with finite abstract simplicial complexes; If K is such a complex we simply say that a *k-face* is a face in K of dimension k. In particular 0-dimensional, 1-dimensional, and 2-dimensional faces are *vertices*, *edges*, and *triangles*, respectively.
Let σ be a nonempty non-maximal face of K. We say that σ is *free* if it is contained in only one maximal face τ of K. Let K' be the simplicial complex obtained from K by removing σ and all faces above it, that is,

$$K' := K \setminus \{\vartheta \in K : \sigma \subseteq \vartheta\}.$$

Thus K' arises from K by the elementary collapse induced by σ and τ. We say that a complex K *collapses* to a complex L if there exists a sequence of complexes $(K_1 = K, K_2, \ldots, K_{m-1}, K_m = L)$, called a *sequence of elementary collapses*, such that K_{i+1} arises from K_i by an elementary collapse for any $i \in \{1, \ldots, m-1\}$. A simplicial complex K is said to be *collapsible* if it collapses to a point.
Let $(K_1 = K, K_2, \ldots, K_{m-1}, K_m = L)$ be a sequence of elementary collapses. Then it is easy to see that for every $\eta \in K \setminus L$ there is a unique complex K_i such that $\eta \in K_i$ and $\eta \notin K_{i+1}$. So we say that η *collapses* at step i.

[2] A Morse matching (acyclic matching) is said to be perfect if it admits a single critical cell, and this cell is also a point.

Collapsibility with constraints

In further constructions, we will encounter the following situation: given a complex L glued to some other complexes forming a complex M, we will know some collapsing sequence of L and we will want to "use" this sequence for M. This might or might not be possible.

Definition 4.3. *Let M be a simplicial complex and L be a subcomplex of M. We define the **constrain complex** of the pair (M, L) as*

$$\Gamma(M, L) := \{\vartheta \in L : \vartheta \subseteq \eta \text{ for some } \eta \in M \setminus L\}.$$

Clearly, the constrain complex is a subcomplex of L. Now we have a sufficient condition that tells us when a collapsing sequence for L induces a collapsing sequence for the whole complex M.

Lemma 4.4. *Let M be a complex, L a subcomplex of M, and $\Gamma = \Gamma(M, L)$. Assume that L collapses to L' containing Γ. Then M collapses to*

$$M' := L' \cup (M \setminus L).$$

Proof. Let $(L_1 = L, L_2, \ldots, L_{m-1}, L_m = L')$ be a sequence of elementary collapses. Let σ_i be the face of L_i which is collapsed in order to obtain L_{i+1} and let τ_i be the unique maximal face in L_i containing σ_i. We set $M_i = L_i \cup (M \setminus L)$. The assumption ensure us that all faces in M_i containing σ_i belong to L_i. Therefore $(M_1 = M, M_2, \ldots, M_{m-1}, M_m = M')$ is a sequence of elementary collapses. □

Collapsibility in codimension 1

Here we prove that collapsibility in codimension 1 is polynomial-time solvable; this is only a complementary result, and it is not needed in further proofs. For this purpose, we need the following proposition. It implies that we can collapse an input d-complex K greedily[3], and with this algorithm, we obtain a $(d-1)$-complex if and only if K collapses to a $(d-1)$-complex L.

Proposition 4.5. *Let K be a d-complex which collapses to a $(d-1)$-complex L and to some d-complex M. Then M collapses to a $(d-1)$-complex.*

[3]An algorithm is said to be *greedy* if it locally makes the optimal choice at each step; see Cormen [1, Chapter 15].

Proof. Let $(K_1 = K, K_2, \ldots, K_{m-1}, K_m = L)$ be a sequence of elementary collapses, where the collapse from K_i to K_{i+1} is induced by faces σ_i and τ_i; since L is a $(d-1)$-complex, without loss of generality we can assume $\dim \tau_i = d$ and $\dim \sigma_i = d-1$ for all i. We observe that every d-dimensional face of K is τ_i for some i. If M is already a $(d-1)$-complex we conclude, otherwise let j be the smallest index such that τ_j belongs to M; we also observe that no τ_i with $i < j$ belongs to M as a consequence of our choices. Now, since σ_j is free in K_j, the only d-faces of K containing σ_j might be τ_i with $i \leq j$. Thus, τ_j is the unique d-face of M containing σ_j, so σ_j can be collapsed. If M is still d-dimensional, we repeat the procedure, and after finitely many steps we obtain a $(d-1)$-complex. □

4.1 Approaching the problem

In this section we describe the approach of Malgouyres and Francés[6], and we introduce auxiliary constructions; further, we will explain how Tancer [7] follows their approach modifying some steps in order to obtain his result.

The reduction is done from the 3-satisfiability problem. Given a 3-CNF formula Φ, Malgouyres and Francés construct a 3-dimensional complex $\mathcal{C}(\Phi)$ such that $\mathcal{C}(\Phi)$ collapses to a 1-complex if and only if Φ is satisfiable. They compose $\mathcal{C}(\Phi)$ of several smaller complexes that we call **gadgets**.

For every literal ℓ in the formula they introduce a *literal gadget* $\mathcal{C}(\ell)$. The gadgets $\mathcal{C}(\ell)$ and $\mathcal{C}(\bar{\ell})$ ($\bar{\ell}$ is the negation of ℓ) are glued along an edge so that a major part of only $\mathcal{C}(\ell)$ or only $\mathcal{C}(\bar{\ell})$ can be collapsed in the first phase of collapsing. Another gadget is the *conjunction gadget* $\mathcal{C}_{\text{and}}$ glued to literal gadgets via *clause gadgets* so that $\mathcal{C}_{\text{and}}$ can be collapsed if and only if every clause contains a literal ℓ such that the major part of $\mathcal{C}(\ell)$ was already collapsed, that is, if and only if Φ is satisfiable. As soon as $\mathcal{C}_{\text{and}}$ is collapsed, it makes few other faces free allowing the whole complex to collapse to a 1-dimensional one. Now this resulting complex contains many cycles, thus it cannot be collapsed to a point.

Intuitively, our idea relies on filling the cycles of the resulting 1-complex so that we can proceed with other collapsings. However, we cannot fill the cycles in an obvious way, since we do not know a priori which complex we obtain. Additionally, filling these cycles could potentially introduce other collapsing sequences which could yield to collapsing the complex even if the formula were not satisfiable.

We are going to construct a simplicial complex $K(\Phi)$ which collapses to a point if

and only if Φ is satisfiable[4]. We reuse literal and conjunction gadgets of Malgouyres and Francés; we need to replace, though, their simple clause gadget. For this construction we need to introduce Bing's house, and we also need *disk gadgets* to fill the cycles in the resulting 1-complex (despite the name, they will not be topological disks but only some contractible complexes).

Bing's Rooms and Bing's houses

Bing's house is a simplicial complex obtained by gluing two smaller complexes called Bing's rooms (depicted in fig. 4.1). *Bing's room with a thin wall* is a complex containing only 2-dimensional faces, whereas *Bing's room with a thick wall* contains one 3-dimensional block obtained by thickening one of the walls. Both rooms contain two holes in the ground floor and one hole in the roof. If now, starting from the room with a thick wall, we collapse[5] away the thick wall, we obtain *Bing's room with a collapsed thick wall*. Note that in fig. 4.1, on the right, the left bottom edge of the collapsed wall is still present.

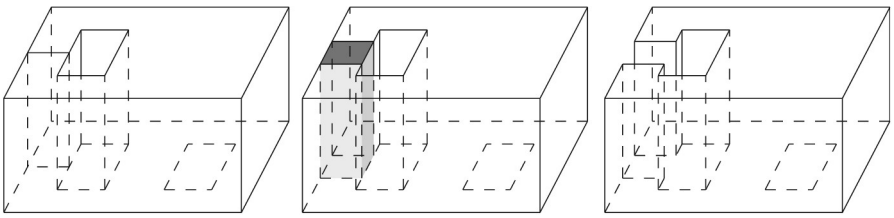


Figure 4.1: *Bing's rooms*

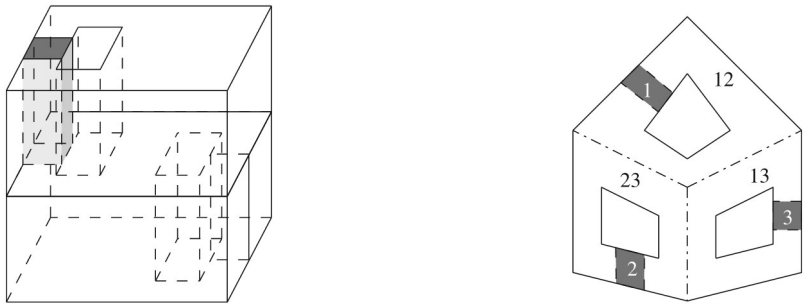
If we rotate the ground floor of one of the rooms and we glue the two rooms together along the ground floor, we obtain *Bing's house with one thin and one thick wall* (fig. 4.2a). Similarly, we can obtain Bing's house with two thin walls or Bing's house with two thick walls.

We also need to introduce Bing's house with three rooms. First we consider the *base floor* as in fig. 4.2b; it consists of three squares with holes (assume that also these holes are square), glued together. Now we consider three Bing's rooms with thick walls labeled 1, 2, and 3. The room labeled i is glued to the two squares with label i so that the gray part of one of the squares labeled i is the place where the thick wall of the room is glued to the base floor.[6]

[4]In fact, $K(\Phi)$ will always be contractible (not in a natural way), but we do not need this fact.

[5]This is not only a cells removal, but an actual collapse in terms of Discrete Morse Theory.

[6]We do not have to distinguish whether the rooms are glued to the base floor from below or from above, since we could not place them in such a way simultaneously in 3D.



(a) *Bing's house with two rooms* **(b)** *Base floor*

Figure 4.2

The resulting complex is the ***Bing's house with three rooms*** *(and three thick walls).* This complex is contractible, and it can be shown in a similar way as the contractibility of classical Bing's house.

It will also be convenient to work with Bing's house with three rooms where the thick walls are collapsed; in particular, we let collapse each of them to the edge on the base floor, and we obtain *Bing's house with three collapsed walls.* In fig. 4.3 we provide an example with two rooms only; edge x_i is the only remaining edge of the thick wall in room i after collapsing the wall. Observe that x_1, x_2, and x_3 are the only free faces of Bing's house with three collapsed walls; see fig. 4.7 for the base floor.

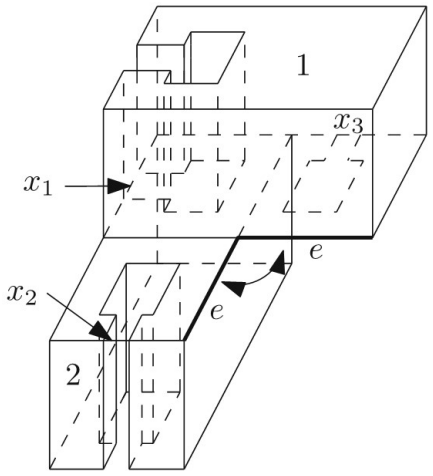
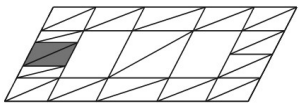


Figure 4.3: *Two blocks of Bing's house with three collapsed walls. The edges marked with e are glued together.*

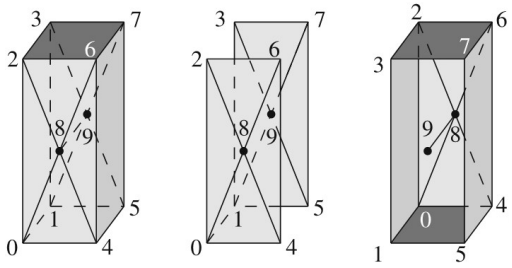
Now, in order to obtain simplicial complexes we triangulate our gadgets. It will not be important how precisely we triangulate pieces of dimension 2 or less. An example of triangulation for the middle level of Bing's house with one thick and one thin wall is shown in fig. 4.4a; we only need to keep in mind that triangulations have to be compatible

with intersections. Also, in some cases we will need gadgets with many prescribed edges in some part of the triangulation, with the number of these edges depending on the size of the 3-CNF formula; in such cases we require that the size of the triangulation is polynomial in the number of the prescribed edges.

Thick walls are the only 3-cells appearing in our constructions, and for these cells we use triangulations of Malgouyres and Francés (fig. 4.4b); in particular, the thick wall is subdivided into four prisms 012389, 014589, 236789, and 456789. Then, each prism is subdivided into two simplices (which are not shown in the picture). This triangulation allows collapsing the thick wall into a smaller complex; for example, the middle picture in fig. 4.4b shows a collapsing used when obtaining Bing's room with a collapsed thick wall from Bing's room with a thick wall.



(a) *A suitable triangulation*



(b) *Collapsings of the thick wall*

Figure 4.4

4.2 The Reduction Gadgets

Here we show details of the construction described at the beginning of the chapter: given a 3-CNF formula Φ, we construct a 3-dimensional simplicial complex $K(\Phi)$. We assume that every clause contains exactly three literals and that no clause contains a literal and its negation.

This complex will consists of several gadgets. For each of them we also need to find some appropriate collapsing sequence (we usually postpone the proof of the existence of such a sequence so that the technical details are left to the end).

Literal gadget

First we recall the *literal gadget* $K(\ell, \bar{\ell})$ (by Malgouyres and Francés) for every pair of literals ℓ and $\bar{\ell}$, then we glue it to other gadgets in a different way. It consists of two smaller parts $X(\ell)$ and $X(\bar{\ell})$ glued together in a suitable way.

Set $X(\ell)$ to be Bing's house with two thick walls; see fig. 4.5. It contains two distinguished edges $e(\ell)$ and $f(\ell)$, and it also contains a path $p(\ell)$ joining their common vertex with the upper thick wall.[7] Recall that we are using the particular triangulation of the upper thick wall described in Section 4.1; the path $p(\ell)$ enters the upper wall in vertex 0 and it continues to vertex 8. Analogously, we construct $X(\bar{\ell})$.

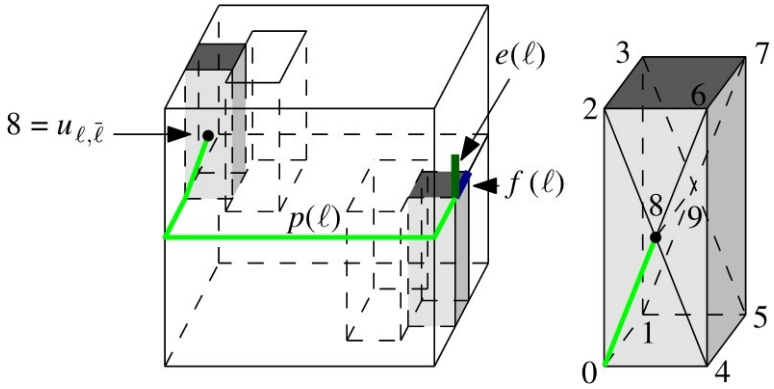


Figure 4.5: *The complex $X(\ell)$ from the literal gadget.*

The gadget $K(\ell, \bar{\ell})$ is obtained gluing $X(\ell)$ and $X(\bar{\ell})$ together along edge 89. We rename the common vertex 8 to $u_{\ell,\bar{\ell}}$, since it will be significant for further constructions. Now, we have the following lemma describing a sequence of collapses for the literal gadget. The statement also says that at least one of $f(\ell)$, $f(\bar{\ell})$ has to be collapsed before collapsing the whole gadget to a 2-complex.

Lemma 4.6.

(1) $K(\ell, \bar{\ell})$ collapses to a complex that contains only $p(\ell)$, $e(\ell)$ and $f(\ell)$ from $X(\ell)$ while it contains almost all $X(\bar{\ell})$ except the upper thick wall, which was collapsed to a thin wall; see fig. 4.4b right, removing edge 89.

(2) Let $L(\ell, \bar{\ell})$ be the complex resulting in item 1 without the edge $e(\ell)$. This complex further collapses to the union of the paths $p(\ell)$, $p(\bar{\ell})$ and the edge $e(\bar{\ell})$.

(3) Let $T(\ell)$ be any of the two triangles containing $e(\ell)$ and $T(\bar{\ell})$ be any triangle containing $e(\bar{\ell})$. Before collapsing both $T(\ell)$ and $T(\bar{\ell})$, at least one of the edges $f(\ell)$, $f(\bar{\ell})$ must be collapsed.

Item 3 is already proved by Malgouyres and Francés [6]. We sketch here that if neither $f(\ell)$ nor $f(\bar{\ell})$ is collapsed, then only one of the two upper thick walls can be collapsed in order to make its edge 01 free. Thus, only one of the triangles might become free before collapsing $f(\ell)$ or $f(\bar{\ell})$. We postpone the proof of items 1 and 2 to Section 4.4.

[7] This path contains neither $e(\ell)$ nor $f(\ell)$.

Conjunction gadget

Now we define the *conjunction gadget* K_{and}. In $K(\Phi)$, instead of having a gadget for every single conjunction, we have a single copy K_{and} representing them all.
This gadget is Bing's house with one collapsed thick wall and one thin wall with several distinguished edges and vertices. See fig. 4.6 on the left.
For every pair ℓ, $\bar{\ell}$ of literals, we create an anchor-shaped tree $A(\ell, \bar{\ell})$ composed of $u_{\ell,\bar{\ell}}$, $p(\ell)$, $p(\bar{\ell})$, $f(\ell)$ and $f(\bar{\ell})$ from $K(\ell, \bar{\ell})$ and also of newly introduced edge $a(\ell, \bar{\ell})$ and vertex v_{and}. See fig. 4.6 on the right. Next we glue all trees $A(\ell, \bar{\ell})$ in vertex v_{and} obtaining a tree A.

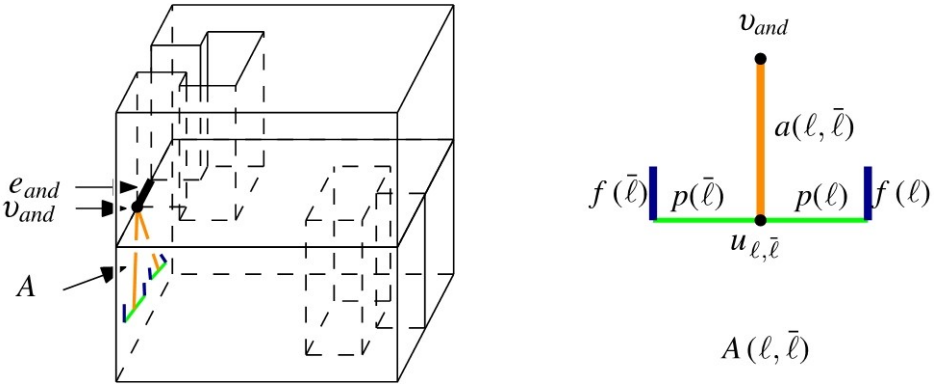


Figure 4.6: *Conjunction gadget*

Finally, we denote with e_{and} the only free edge of K_{and} and we glue A to the lower left wall of K_{and}. Note that every literal gadget is glued to the conjunction gadget.
As soon as we have introduced all gadgets, we will see that K_{and} is glued to other gadgets only along A and e_{and}. The following lemma states that collapsing K_{and} needs e_{and} to be free first in whole $K(\Phi)$ and only then we can continue with collapsing K_{and}. On the other hand, once we make e_{and} free, we can collapse the complex to A.

Lemma 4.7.

(1) K_{and} *collapses to* A.

(2) Before collapsing any triangle containing one of $f(\ell)$, $f(\bar{\ell})$, *the edge* e_{and} *has to be collapsed*

We prove item 1 in Section 4.4; item 2 is explained in [6].

Clause gadget

Next we introduce the *clause gadget*: for a clause $c = (\ell_1 \vee \ell_2 \vee \ell_3)$ we set $K(c)$ to be Bing's house with three collapsed walls with several distinguished edges and paths.

In particular, the only three free edges of $K(c)$ are labeled (ℓ_i, c); see fig. 4.7. We also label three paths $p(\ell_i, c)$ connecting the center of the base floor with (ℓ_i, c) (we assume that the path do not contain (ℓ_i, c)). Finally, we distinguish one other edge, labeled e_{and}, going from the center inside the base floor. This last edge is glued together with the edge of the conjunction gadget with the same label so that the central vertex of the base floor becomes the vertex v_{and}. We have the following facts.

Lemma 4.8.

(1) $K(c)$ collapses to a complex composed of e_{and}, three paths $p(\ell_i, c)$, and two of the three edges (ℓ_i, c).

(2) Any collapsing of $K(c)$ starts with one of the edges (ℓ_i, c).

Item 2 easily follows from the fact that the only free faces of $K(c)$ are the three edges (ℓ_i, c). The proof of item 1 is rather postponed to Section 4.4.

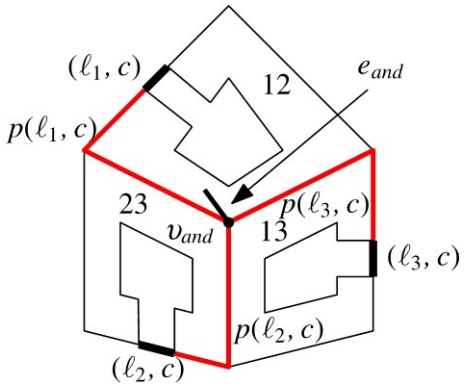


Figure 4.7: *Base floor of the clause gadget*

Disk gadget

Lastly, for every pair of literals ℓ, $\bar{\ell}$ we construct the *disk gadget* $D(\ell, \bar{\ell})$. We first consider Bing's house with one collapsed thick wall and one thin wall; see fig. 4.8a. We label the only free face of this complex with $e(\ell)$ and glue it to the edge $e(\ell)$ of $K(\ell, \bar{\ell})$. Now we choose a vertex on the edge connecting the left an the bottom wall and label it v_{and}. We also glue this vertex to the v_{and} of the conjunction gadget. The edge connecting v_{and} and one of the vertices of $e(\ell)$ is labelled by $b(\ell)$. Next, for every clause c_j containing ℓ we make a copy of $p(\ell, c_j)$ and (ℓ, c_j), starting in vertex v_{and}. In particular, the complex described above is glued to the complexes $K(c_j)$ along these paths and edges. We denote the resulting complex with $B(\ell)$. For $B(\bar{\ell})$ we proceed analogously.

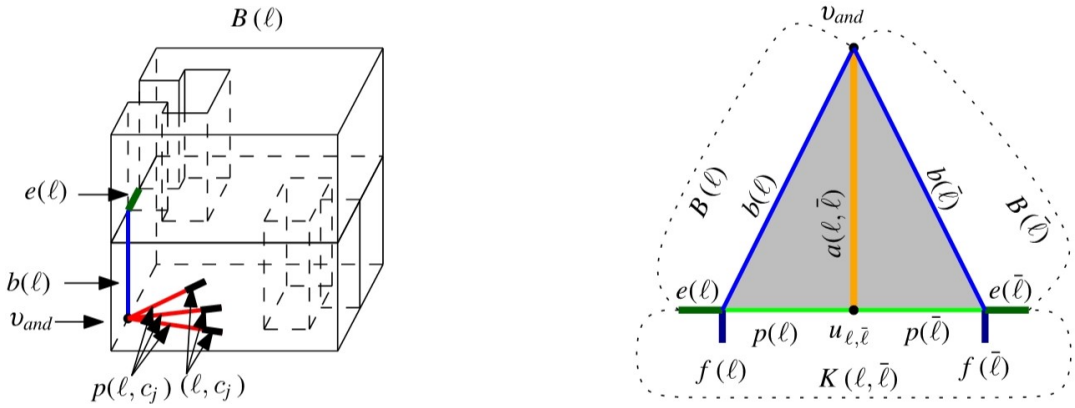
The following lemma ensures that if the edge $e(\ell)$ become free, then this complex can be collapsed (inside whole $K(\Phi)$) to a complex composed of the distinguished edges and paths.

Lemma 4.9.

(1) $B(\ell)$ collapses to the 1-complex composed of $b(\ell)$, paths $p(\ell, c_j)$, and edges (ℓ, c_j).

(2) Any collapsing of $B(\ell)$ starts with the edge $e(\ell)$.

As usual, item 1 is proved in Section 4.4; item 2 follows as in the previous lemma.



(a) *Complex $B(\ell)$* **(b)** *Disk gadget*

Figure 4.8

Now we can construct $D(\ell, \bar{\ell})$ by filling two cycles with a disk (fig. 4.8b). The first cycle is formed by $b(\ell)$, $p(\ell)$ and $a(\ell, \bar{\ell})$, the second one by $b(\bar{\ell})$, $p(\bar{\ell})$ and $a(\ell, \bar{\ell})$. Now the construction of $D(\ell, \bar{\ell})$ is finished and we have already described all gluings, so this concludes the construction of $K(\Phi)$.

4.3 Collapsibility of $K(\Phi)$

In this section we prove Theorem 4.1 showing that $K(\Phi)$ is collapsible if and only if Φ is satisfiable.

Satisfiable formulas

First we assume that Φ is satisfiable, and we also fix one satisfying assignment. We want to construct a collapsing sequence for $K(\Phi)$ proceeding in several steps. By $K^{(i)}(\Phi)$ we denote the complex obtained after performing i-th step of collapsing. We also denote gadgets in a similar way, for example, $K^{(i)}_{\text{and}}$ is $K^{(i)}(\Phi) \cap K_{\text{and}}$.

1. For every literal ℓ we "partially" collapse $K(\ell, \bar{\ell})$ as in Lemma 4.6. Note that the constrain complex of $(K(\Phi), K(\ell, \bar{\ell}))$ consists of $p(\ell)$, $p(\bar{\ell})$, $e(\ell)$, $e(\bar{\ell})$, $f(\ell)$ and $f(\bar{\ell})$; therefore Lemma 4.4 induces collapsing on whole $K(\Phi)$. Now if ℓ has positive occurrence in the assignment, we let $X(\ell)$ collapse to $p(\ell)$, $e(\ell)$ and $f(\ell)$ while in $X(\bar{\ell})$ only its upper thick wall collapses to a thin one. After this, the edge $e(\ell)$ is free.

   Once we have gradually performed this collapsing for all literals with positive occurrence, we do not "miss" negative ones since for every input variable u exactly one literal of u and $\neg u$ has positive occurrence.

2. Next we collapse $B(\ell)$ as stated in Lemma 4.9. At this step, the constrain complex of $(K^{(1)}(\Phi), B^{(1)}(\ell)) = (K^{(1)}(\Phi), B(\ell))$ contains only $b(\ell)$, paths $p(\ell, c_j)$, and edges (ℓ, c_j), thus collapsing from Lemma 4.9 induces collapsing of $K^{(1)}(\Phi)$ by Lemma 4.4. From now on we will use this lemma many other times without mentioning it.

3. Now for every clause c at least one of the edges (ℓ_i, c) became free, since the assignment is satisfying. Then, every clause gadget collapses to the 1-complex described in Lemma 4.8. The constrain complex of $(K^{(2)}(\Phi), K^{(2)}(c))$ is a subcomplex of the complex formed by paths $p(\ell_j, c)$ and edges (ℓ_j, c), with $j \neq i$.

4. Here we focus on the edge e_{and}; at the beginning, it was contained in triangles of clause gadgets and in a single triangle of the conjunction gadget. All triangles of clause gadgets were collapsed in previous steps, so e_{and} is now free. Lemma 4.7 allows us to collapse K_{and} to A, and also the constrain complex of $(K^{(3)}(\Phi), K^{(3)}_{\text{and}})$ is A.

5. In this step, we will collapse literal and disk gadgets. The crucial fact is that the edges $f(\ell)$ and $f(\bar{\ell})$ are already free, thus Lemma 4.6 allows us to proceed with collapsing $K^{(4)}(\ell, \bar{\ell})$. Now $e(\bar{\ell})$ is free as well as all remaining edges of $p(\ell)$ and $p(\bar{\ell})$, so we can easily collapse $B(\bar{\ell})$ thanks to Lemma 4.9 and consequently also $D(\ell, \bar{\ell})$.

6. In the end, we have a collection of paths starting from v_{and} that can be easily collapsed to v_{and} itself.

Non-satisfiable formulas

To conclude, we show that $K(\Phi)$ is not collapsible when Φ is not satisfiable.
Suppose that $K(\Phi)$ is collapsible, then in particular some triangle of K_{and} has to be

collapsed at some step. Consider the first step in which such a triangle has to be collapsed. Now, according to Lemma 4.7, the edge e_{and} has to be made free before this step. By Lemma 4.8 for every clause c there is a literal $\ell(c)$ in it such that the edge $(\ell(c), c)$ was made free prior this step, thus by Lemma 4.9 the edge $e(\ell(c))$ had to be made free previously. Recall now that no triangle of K_{and} was collapsed yet, therefore only one of the edges $e(\ell)$ and $e(\bar{\ell})$ can be collapsed at this stage, according to Lemma 4.6. This leads us to a satisfying assignment for Φ: set a variable u to TRUE if $e(u)$ was collapsed, FALSE otherwise.

4.4 Collapsing Sequences

Here we prove technical lemmas used in the previous sections (the order of proofs is changed for convenience).

Proof of Lemma 4.7(1). Recall that we want to collapse the conjunction gadget to the tree A; see fig. 4.6. First, we collapse the wall below e_{and} and then the lowest floor (except edges in A). Next we collapse all walls that were attached to the lowest floor. After this, we obtain the complex depicted in fig. 4.9: this is a 2-sphere with a hole and with A attached to it. It is easy to conclude the collapsing sequence (proceeding in the direction of the arrows). □

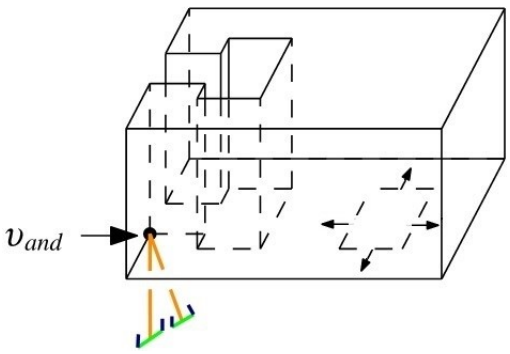


Figure 4.9: *An intermediate step in collapsing K_{and}.*

Proof of Lemma 4.8(1). In order to collapse the clause gadget (fig. 4.7), we will provide a collapsing to the union of paths $p(\ell_i, c)$ and edges (ℓ_2, c), (ℓ_3, c); other cases are analogous. We will assume that Bing's room number 1 is above the base floor and that the number 2 is below the base floor as in fig. 4.10 on the left.
Since (ℓ_1, c) is allowed to be collapsed, we can collapse the left wall of room 2 and then the bottom one. Next, we can collapse all walls of room 2 perpendicular to the base

floor, so we obtain a complex as in fig. 4.10 on the right. After this, we collapse the interior of the 23 square so that left edges of (ℓ_2, c) become free. Then the room 3 can be collapsed in a similar way as we did with room 2; note that, after this step, only (ℓ_2, c), $p(\ell_2, c)$ and part of $p(\ell_1, c)$ remain of the 23 square. Finally, we can collapse room 1 in a similar fashion taking care that e_{and} be left uncollapsed. □

Proof of Lemma 4.6(1). First, we collapse the thick wall of $X(\bar{\ell})$ as shown in fig. 4.4b, on the right. After this the edge 89, in common with $X(\ell)$, becomes free. Then we can collapse the thick wall of $X(\ell)$ so that the upper Bing's room of it becomes Bing's room with collapsed thick wall. At this point, the rest of $X(\ell)$ can be collapsed in the same way as in the proof of Lemma 4.7 while keeping $p(\ell)$ and $f(\ell)$. □

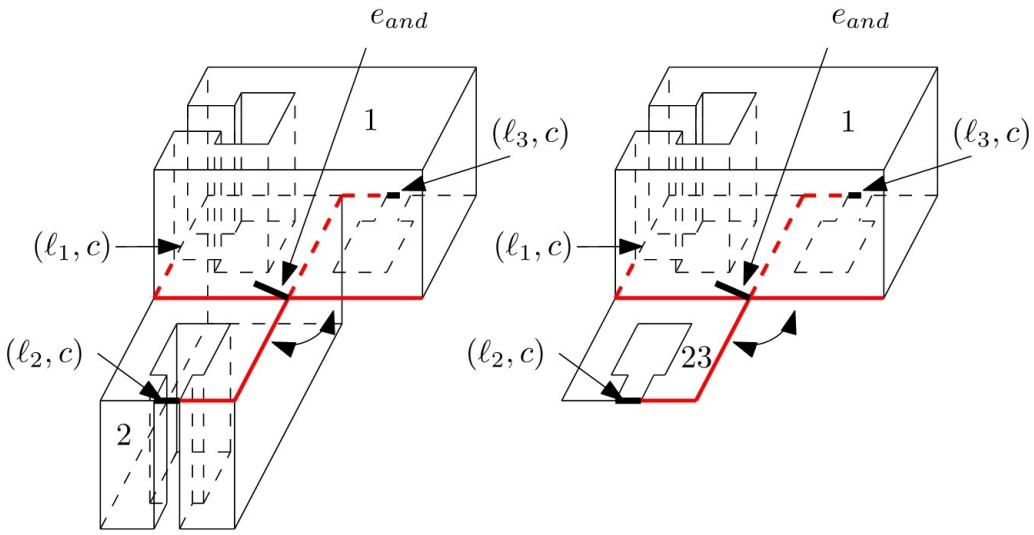


Figure 4.10: *Rooms of the clause gadget while collapsing it*

Proof of Lemma 4.6(2). As soon as we are allowed to collapse $f(\bar{\ell})$, the lower thick wall of $X(\bar{\ell})$ can be collapsed obtaining Bing's room with collapsed thick wall from the lower room. Now we can collapse $X(\bar{\ell})$ in similar way as in the proof of Lemma 4.7 while keeping the prescribed subcomplex; clearly, the roles of the lower and upper rooms are interchanged. □

Proof of Lemma 4.9(1). Here we use the same collapsing procedure as in the proof of Lemma 4.7. We just remark that the wall below $e(\ell)$, split by $b(\ell)$, is collapsed in two stages. First we collapse the half containing $e(\ell)$; then the lowest floor of $B(\ell)$ is collapsed, and we finish collapsing the second half of this wall. □

4.5 Conclusion

We have shown that it is NP-complete to decide whether a 3-dimensional complex is collapsible, that is, it collapses to a point. Here we mention a few corollaries of this result, and we also analyze the remaining cases.

We set up question (d, k)-Collapsibility asking whether a given d-dimensional complex collapses to some k-dimensional one, where $d > k \geq 0$ are fixed.
The above result shows that $(3, 0)$-Collapsibility is NP-complete; moreover, observe that it is not difficult to extend it showing that (d, k)-Collapsibility is NP-complete for any $d \geq 3$ and $k \in \{0, 1\}$. For this we only need to attach a d-simplex to v_{and} and remark that if Φ is not satisfiable, then any collapsing of $K(\Phi)$ yields a complex of dimension 2 or more (we also remark that the case $d \geq 3$ and $k = 1$ has been already examined from the construction of Malgouyres and Francés [6]).[8]
As we mentioned before in the chapter, it is not hard to prove polynomial-time solvability of (d, k)-Collapsibility whenever $d \leq 2$, and also in the codimension 1 case; see Proposition 4.5.

Answering the question completely

In the remaining cases, when $d \geq k + 2$ $(d \neq 2)$, it is reasonable to believe that the problem is also NP-complete. In fact, we now extend previous results with the following theorem, proved by Paolini [8].

Theorem 4.10. *The* (d, k)-Collapsibility *problem is NP-complete for* $d \geq k + 2$*, except for the case* $(2, 0)$.

For this purpose, in Theorem 4.11 we show a polynomial-time reduction of (d, k)-Collapsibility to $(d + 1, k + 1)$-Collapsibility; the main result will then follow by induction, with the base cases given by NP-completeness of $(3, 1)$-Collapsibility and $(d, 0)$-Collapsibility $(d \geq 3)$.
Recall that "collapsibility to some k-dimensional subcomplex" is equivalent to "existence of an acyclic matching such that the critical cells form a k-dimensional subcomplex", see Chapter 2. Also observe that, given an acyclic matching with no critical simplices of dimension $d > k$, one can always remove from it the arcs between its simplices obtaining a new acyclic matching where the critical simplices form now a subcomplex of dimension k.

[8] Similarly, given a 1-complex we can attach it to v_{and}, then our 3-dimensional complex collapses to this fixed 1-complex. This shows that it is NP-complete to decide whether a 3-dimensional complex collapses to a fixed 1-complex.

Theorem 4.11. *Let $d > k \geq 0$. Then there is a polynomial-reduction from (d,k)-*Collapsibility *to $(d+1,k+1)$-*Collapsibility*.*

Proof. Let X be an instance of (d,k)-Collapsibility, that is, a d-dimensional simplicial complex. Let $V = v_1, \ldots, v_r$ be the vertex set of X. Now we construct an instance X' of $(d+1,k+1)$-Collapsibility as follows.
Let n be the number of simplices in X. Introduce new vertices $w_1, \ldots, w_{n+1}$, and define X' as the simplicial complex on the vertex set $V' = v_1, \ldots, v_r, w_1, \ldots, w_{n+1}$ given by

$$X' = X \cup \Big\{\sigma \cup \{w_i\} \mid \sigma \in X, i = 1, \ldots, n+1\Big\}.$$

Note that this is the $(n+1)$-coning introduced in Section 1.2. Thus X' has $n(n+2)$ simplices. Broadly speaking, we obtain X' from X by attaching $n+1$ cones of X to it. We now prove that X is a yes-instance of (d,k)-Collapsibility if and only if X' is a yes-instance of $(d+1,k+1)$-Collapsibility.

Suppose first that X is a yes-instance of (d,k)-Collapsibility. Then there exists an acyclic matching $\mathcal{M}$ on X such that all critical simplices have dimension at most k. We construct a matching $\mathcal{M}'$ on X' as follows:

$$\begin{aligned}\mathcal{M}' = &\Big\{(\sigma, \sigma \cup \{w_1\}) \mid \sigma \in X\Big\} \\ &\cup \Big\{(\tau \cup \{w_i\}, \sigma \cup \{w_i\}) \mid (\tau,\sigma) \in \mathcal{M}, i = 2, \ldots, n+1\Big\}.\end{aligned} \tag{4.1}$$

In terms of collapses, this matching correspond to collapsing the first cone together with X, and every other "base-less" cone by itself (as we are collapsing X). Note that the critical simplices of $\mathcal{M}'$ do not form a subcomplex of X', even when the critical simplices of $\mathcal{M}$ form a subcomplex of X.

Now we prove that $\mathcal{M}'$ is acyclic by considering the set $P = \{w_1, \ldots, w_{n+1}\}$ with the partial order

$$w_i < w_j \text{ if and only if } i = 1 \text{ and } j > 1.$$

Let $\varphi : X' \to P$ be the poset map given by

$$\varphi(\sigma) = \begin{cases} w_j & \text{if } \sigma \text{ contains } w_j \text{ for some } j \geq 2; \\ w_1 & \text{otherwise.} \end{cases}$$

Thus $\mathcal{M}'$ is a union of matchings $\mathcal{M}'_j$ on each fiber $\varphi^{-1}(w_j)$. The arcs of $\mathcal{M}'_1$ define a cut of the Hasse diagram of $\varphi^{-1}(w_1)$, then the matching $\mathcal{M}'_1$ is acyclic on $\varphi^{-1}(w_1)$. The Hasse diagram of each $\varphi^{-1}(w_j)$ for $j \geq 2$ is isomorphic to the Hasse diagram of $X \cup \{\emptyset\}$ via the map $\sigma \cup \{w_j\} \mapsto \sigma$, and the matching $\mathcal{M}'_j$ maps to $\mathcal{M}$. Then each $\mathcal{M}'_j$ is acyclic

on $\varphi^{-1}(w_j)$, since $\mathcal{M}$ is acyclic on the Hasse diagram of X.
By the Patchwork Theorem 2.13, $\mathcal{M}'$ is acyclic on X'.

The set of critical simplices of $\mathcal{M}'$ is

$$\mathrm{Cr}(X', \mathcal{M}') = \{w_1\} \cup \Big\{\sigma \cup \{w_i\} \mid \sigma \in \mathrm{Cr}(X, \mathcal{M}) \cup \{\emptyset\}, i = 2, \dots, n+1\Big\}.$$

In particular, all critical simplices have dimension at most $k+1$. Therefore X' is a yes-instance of $(d+1, k+1)$-Collapsibility.

Vice versa, suppose now that X' is a yes-instance of $(d+1, k+1)$-Collapsibility. Thus we have an acyclic matching $\mathcal{M}'$ on X' such that all critical simplices have dimension at most $k+1$. X contains n simplices, and there are $n+1$ cones, then there exists an index $j \in \{1, \dots, n+1\}$ such that

$$\Big(\sigma, \sigma \cup \{w_j\}\Big) \notin \mathcal{M}' \text{ for all } \sigma \in X.$$

That is, simplices containing w_j are only matched with simplices containing w_j. Thus we can construct a matching $\mathcal{M}$ on X as follows:

$$\mathcal{M} = \Big\{(\tau, \sigma) \mid \tau, \sigma \in X \text{ such that } \Big(\tau \cup \{w_j\}, \sigma \cup \{w_j\}\Big) \in \mathcal{M}'\Big\}.$$

Note that if there is some 0-dimensional simplex $\sigma = \{v\} \in X$ such that $(\{w_j\}, \{v, w_j\}) \in \mathcal{M}'$, then $\{v\} \in \mathrm{Cr}(X, \mathcal{M})$. The Hasse diagram of X injects into the one of the j-th cone via the map $\sigma \mapsto \sigma \cup \{w_j\}$, and so $\mathcal{M}$ maps to $\mathcal{M}'$. Therefore $\mathcal{M}$ is also acyclic, and its set of critical simplices is

$$\mathrm{Cr}(X, \mathcal{M}) = \Big\{\sigma \in X \mid \sigma \cup \{w_j\} \in \mathrm{Cr}(X', \mathcal{M}') \text{ or } \Big(\{w_j\}, \sigma \cup \{w_j\}\Big) \in \mathcal{M}'\Big\}.$$

In the first case $\sigma \cup \{w_j\}$ has dimension at most $k+1$, and in the second one σ is 0-dimensional. In particular, all critical simplices have dimension at most k. Therefore X is a yes-instance of (d, k)-Collapsibility. □

The following examples clarify the constructions described above from a geometrical viewpoint.

Example 4.12. Let X be the simplicial complex depicted in fig. 4.11 on the left, with its associated acyclic matching $\mathcal{M}$ next to it. In this case, using the same notation as in the proof, $n = 3$. Thus we construct the simplicial complex X' from X by attaching 4 cones of X to it; see fig. 4.11 on the right. Now, from $\mathcal{M}$, we define an acyclic matching $\mathcal{M}'$ associated with X'. In fig. 4.12 we show the Hasse diagram of X' with the arrows describing $\mathcal{M}'$: triangular arrows represent the relations given by the first set in (4.1) and classic arrows represent the relations given by the second set.

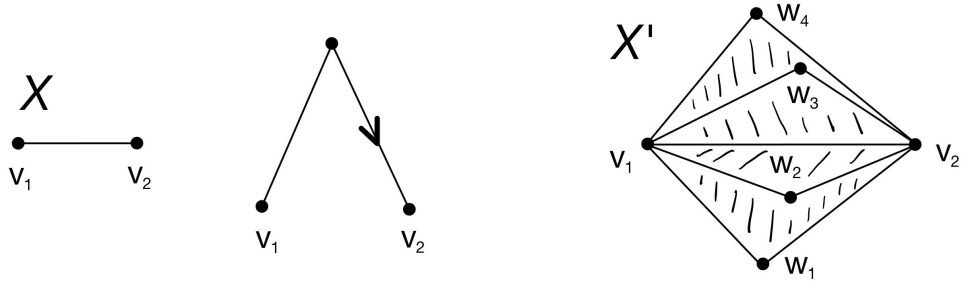


Figure 4.11: *An elementary simplicial complex X with an associated acyclic matching on its Hasse diagram, and the complex resulting from the reduction in the above proof.*

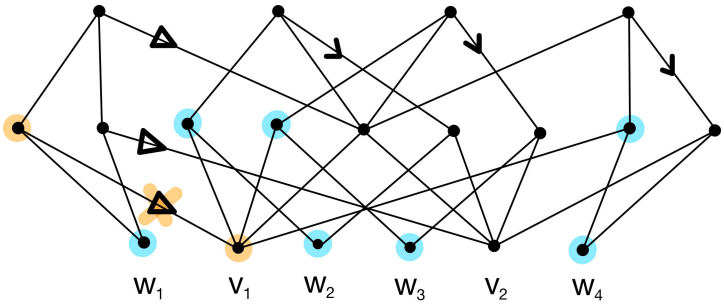


Figure 4.12: *The Hasse diagram of X' with the resulting acyclic matching $\mathcal{M}'$.*

As mentioned before, we can always ignore some matchings in order to make the set of critical simplices a subcomplex of X'. In fig. 4.12 we highlight in light blue the critical simplices given by $\mathcal{M}'$, while we highlight in orange the new critical simplices obtained by removing the corresponding matching. Geometrically, we have the situation depicted in fig. 4.13, in which we describe the collapsing sequence for X' that gives us a 1-complex. Therefore we performed a reduction from $(1,0)$-Collapsibility to $(2,1)$-Collapsibility.

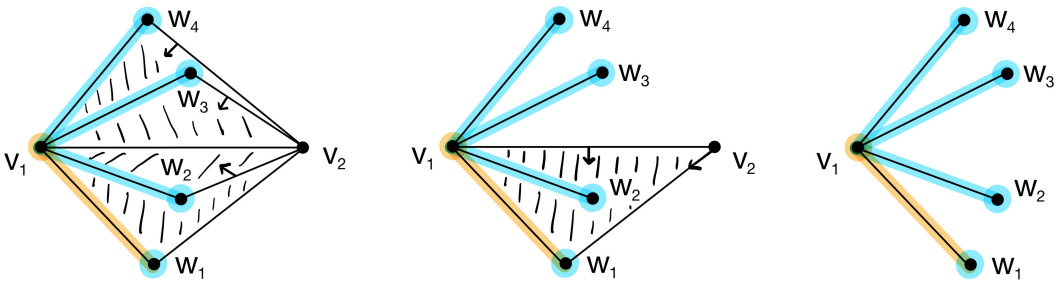


Figure 4.13: *From left to right, the collapsing sequence on X' provided by the matching resulting from the reduction.*

Example 4.13. Let us give as input a more complicated simplicial complex X, like the one in fig. 4.14. Now $n = 10$ and we only describe individually each cone C_i over X forming[9] X', since the obtained complex X' is no more embeddable in $\mathbb{R}^3$; see fig. 4.17. For $i = 2, \ldots, 11$ the acyclic matching $\mathcal{M}'_i$ defined on the Hasse diagram of C_i does not change, and we describe it in fig. 4.15 (arrows and colours have the same meaning already described in the previous example). The acyclic matching $\mathcal{M}'_1$ defined on the Hasse diagram of C_1, meanwhile, differs from the others; it is depicted in fig. 4.16.

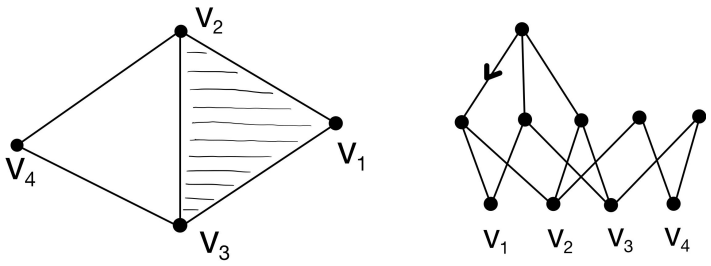


Figure 4.14: *A 2-complex X next to its Hasse diagram with an associated acyclic matching.*

Here these cones are not considered as independent spaces, but we deal with them as components of X'. Hence in the diagram of C_1 we see orange highlights coming from C_i; e.g., if we start from the simplices marked in blue and we want the set of critical simplices to be a subcomplex, then we need to add $\{v_1\}$ to it and so we mark it in orange and consequently we remove the associated matching in C_1. In this example we have a reduction from $(2,1)$-Collapsibility to $(3,2)$-Collapsibility.

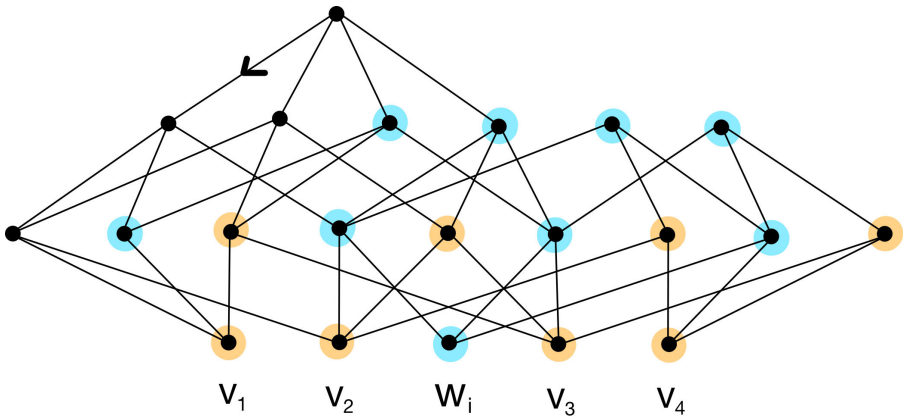


Figure 4.15: *Hasse diagram of C_i*

[9] As described in the proof, the cones are then glued along their "base", that is, the complex X.

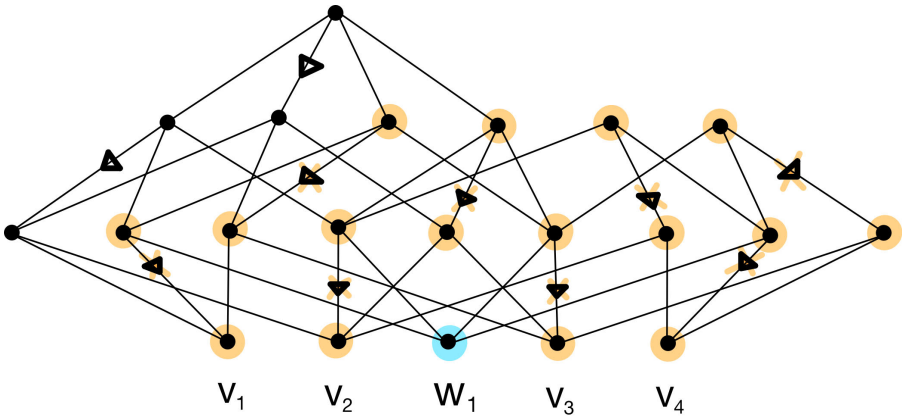


Figure 4.16: *Hasse diagram of* C_1

To sum up, the (d, k)-Collapsibility admits a polynomial-time solution when $d = k + 1$, and also for the case $(2, 0)$ [5, 6, 7]. For $(3, 1)$-Collapsibility we have NP-completeness [6], and in previous sections we show how to extend this result to (d, k)-Collapsibility for $k \in \{0, 1\}$ and $d \geq 3$ [7]. Finally, using this as the base step and Theorem 4.11 as the induction step, we prove Theorem 4.10.

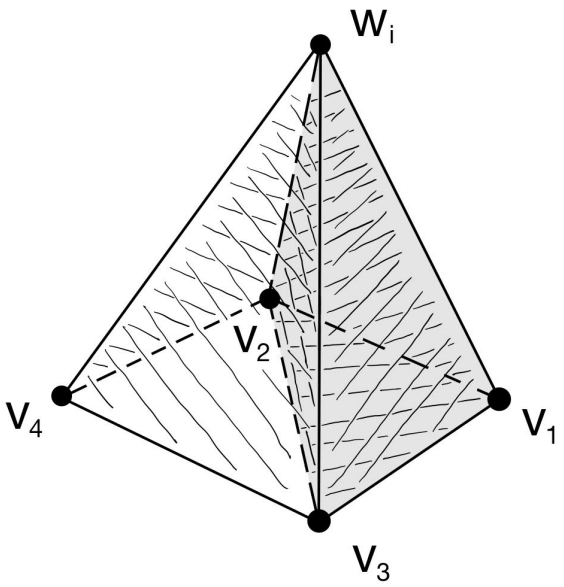


Figure 4.17: *Cone* C_i *over* X

Chapter 5

Appendix

CIRCUIT-SAT

Here follows a proof of Lemma 3.8, that shows NP-hardness of CIRCUIT-SAT.

Proof. Let L be any language in NP. Thus there exist an algorithm A that verifies L in polynomial time. We construct an algorithm F that uses A to compute the reduction function f.
Let $T(n)$ denote the worst-case running time of A on length-n input strings, and let $k \geq 1$ be a constant such that $T(n) = O(n^k)$ and the length of certificate is $O(n^k)$ as well.

The idea of the proof is to represent the computation of A as a sequence of configurations. We consider each configuration as composed by different parts: the program for A, the program counter, the input x, the certificate y, and working storage. The combinational circuit M, which implements the computer hardware, maps each configuration c_i to the next one c_{i+1}, starting from c_0. Algorithm A writes its output to some location when it finishes executing. After A halts, the output value never changes. Thus, if the algorithm runs for at most $T(n)$ steps, the output appears as one of the bits in $c_{T(n)}$.
The reduction algorithm F constructs a single combinational circuit that computes all configurations produced by a given initial configuration. The idea is to connect together $T(n)$ copies of M. The output of the i-th circuit, which produces c_i, feeds into the input of the $(i+1)$st circuit. Thus, the configurations, rather than being stored in memory, simply reside as values on the wires connecting the copies.

More precisely, when F receive an input x, it computes $n = |x|$ and then it constructs a combinational circuit C' consisting of $T(n)$ copies of M. The input to C' corresponds to a computation on $A(x, y)$, and the output is the configuration $c_{T(n)}$. Now, algorithm F modifies C' to construct $C = f(x)$.

First, it wires the inputs to C' corresponding to the program for A, the initial program counter, the input x, and the initial state of these values in memory. Thus, the only remaining inputs correspond to the certificate y.
Second, it ignores all outputs from C', except for the bit of $c_{T(n)}$ corresponding to the output of A. This circuit C, so constructed, computes $C(y) = A(x, y)$ for any input y of length $O(n^k)$. The reduction algorithm F, when provided an input x, computes C and outputs it.

We now must show that F correctly computes f, and that it runs in polynomial time. Let us show that C is satisfiable if and only if there exists y such that $A(x, y) = 1$.
Suppose that there exists a certificate y of length $O(n^k)$ such that $A(x, y) = 1$. Then, once we have applied the bits of y to the inputs of C, the output is $C(y) = 1$. For the other direction, suppose that C is satisfiable. Thus, there exists an input y to C such that $C(y) = 1$, and we conclude that $A(x, y) = 1$.
The number of bits required to represent a configuration is polynomial in n, since the program for A has a constant size, the length of the input x is n, and the length of the certificate y is $O(n^k)$. Since the algorithm runs for at most $O(n^k)$ steps, the amount of storage required by A is polynomial in n as well.
The combinational circuit M implementing the computer hardware has size polynomial in the length of a configuration, and hence, it is polynomial in n. Lastly, the circuit C consists of $O(n^k)$ copies of M, so the size is polynomial in n. □

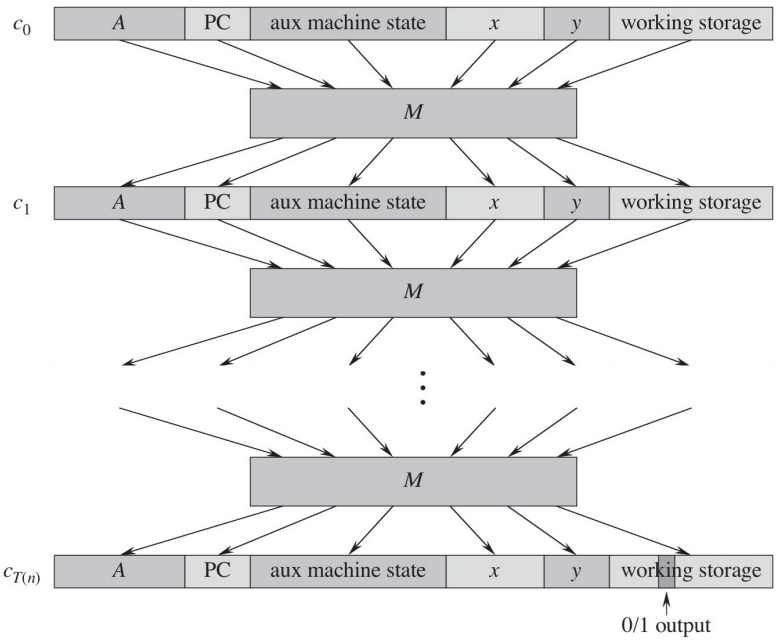


Figure 5.1: *The sequence of configurations produced by an algorithm A running on an input x and certificate y.*

Unrecognizability of Contractible Complexes

We first talk about complexes of dimension at least 5; we only sketch a proof of the following theorem.

Theorem 5.1 (Novikov)**.** *For every $d \geq 5$, it is algorithmically undecidable whether a given simplicial complex of dimension d is contractible.*

The proof of this theorem easily follows from [3, §15].

Sketch of proof. Novikov shows the existence of an efficiently constructible sequence M_j of d-manifolds such that M_j is a ball if and only if $\pi_1(M_j)$ is trivial, and it is algorithmically undecidable whether $\pi_1(M_j)$ is trivial. In order to finish the proof, we need to know that M_j can be efficiently constructed as a simplicial complex. This can be done by inspecting the proof in [3] with not too much effort. □

The dimension 5 in Theorem 5.1 can be dropped to 4, if we greedily collapse 5-dimensional simplices. On the other hand, the contractibility question is trivially polynomial-time solvable for complexes of dimension at most 1, since it is equivalent with recognition of trees. Lastly, regarding complexes of dimension 2 or 3, the question is open to our best knowledge.

Bibliography

[1] T. H. Cormen, *Introduction to Algorithms*, MIT Press, Cambridge, 2022.

[2] D. Kozlov, *Combinatorial Algebraic Topology*, Springer, Berlin, 2008.

[3] I. A. Volodin *et al*, *The problem of discriminating algorithmically the standard three-dimensional sphere*, Russ. Math. Surv. **29** (1974), 71-172.

[4] Ö. Eğecioğlu and T. F. Gonzalez, *A computationally intractable problem on simplicial complexes*, Computational Geometry **6** (1996), no. 2, 85-98. https://doi.org/10.1016/0925-7721(95)00015-1.

[5] M. Joswig and M. E. Pfetsch, *Computing optimal Morse matchings*, SIAM Journal on Discrete Mathematics **20** (2006), 11-25. https://doi.org/10.48550/arXiv.math/0408331.

[6] R. Malgouyres and A. R. Francés, *Determining whether a simplicial 3-complex collapses to a 1-complex is NP-complete*, Discrete geometry for computer imagery, Lecture Notes in Computer Science, vol. 4992, Springer, Berlin, 2008, 177-188. https://doi.org/10.1007/978-3-540-79126-3_17.

[7] M. Tancer, *Recognition of Collapsible Complexes is NP-Complete*, Discrete & Computational Geometry **55** (2016), 21–38. https://doi.org/10.1007/s00454-015-9747-1.

[8] G. Paolini, *Collapsibility to a Subcomplex of a Given Dimension is NP-Complete*, Discrete & Computational Geometry **59** (2017), 246-251. https://doi.org/10.48550/arXiv.1703.06983.

Acknowledgements

I would like to thank my supervisor, Giovanni Paolini, for assisting me with his helpfulness in all that concerns this project.

A special thank goes to my entire family, without whom I would not have made it through this journey. Thank you especially for allowing me to follow my passions and for always encouraging me to do so, by supporting me with all your love and much more. But thank you also for your patience, because I know how much of a procrastinator and how stubborn I can be at times. I promise you that I will keep trying more and more to step out of my comfort zone.

Least but not last, thanks to all my friends, who have helped me in countless ways to enjoy the good times, and especially to face the bad ones. I'm sure we'll continue to have *never-ending fun* moments, like cooking together our favourite dish[1]. I would also like to spend some time fixing electrical sockets during an evening with friends.

[1] *Cappelletto in brodo di fagiano.*