

Levin von Hollen
Tilman Beck

Implementierung der Cut & Count-Technik für das Steiner Tree-Problem

21. November 2016

Springer

Inhaltsverzeichnis

1	Einleitung	1
1.1	Einleitung	1
1.2	Notation	1
2	(Nice) Tree Decomposition	3
2.1	Tree Decomposition	3
2.2	Nice Tree Decomposition	3
2.3	Weitere Modifikationen	5
3	Cut & Count-Technik	7
3.1	Einführendes	7
3.2	Monte-Carlo-Algorithmus	7
3.3	Isolation Lemma	8
4	Cut & Count für Steiner Tree	9
4.1	Steiner Tree	9
4.2	Cut	9
4.3	Count	10
4.4	Dynamisches Programm	10
4.5	Monte-Carlo Algorithmus und Laufzeit	10
5	Implementierung	11
5.1	Nice Tree Decomposition	11
5.2	Dynamisches Programm	12
5.3	Evaluierung	12
5.4	Ausblick	12
6	Zusammenfassung	13
6.1	Text	13

7	Chapter Heading	15
7.1	Section Heading	15
7.2	Section Heading	15
7.2.1	Subsection Heading	16
7.3	Section Heading	17
7.3.1	Subsection Heading	18
	Appendix	20
	Problems	20
	Sachverzeichnis	21
	Literaturverzeichnis	23

Kapitel 1

Einleitung

1.1 Einleitung

1.2 Notation

Für den Rest der Arbeit bedienen wir uns der Notation aus der Arbeit [CNP⁺]. Die Bezeichnung $G = (V, E)$ beschreibt einen ungerichteten Graphen. Entsprechend beschreiben $V(G)$ und $E(G)$ die Menge der Knoten bzw. Kanten des Graphen G . Die Bezeichnung $G[X]$ einer Knotenmenge $X \subseteq V(G)$ steht für den Subgraphen, der von X erzeugt wird. Für eine Menge an Kanten $X \subseteq E$ beschreibt $V(X)$ die Menge der Endknoten der Kanten aus X und $G[X]$ den Subgraphen (V, X) . Die Knotenmenge für eine Menge von Kanten X im Graphen $G[X]$ ist dieselbe wie im Graphen G .

Mit einem „Schnitt“ einer Menge $X \subseteq E$ ist das Paar (X_1, X_2) mit den Eigenschaften $X_1 \cap X_2 = \emptyset, X_1 \cup X_2 = X$. X_1 und X_2 werden als linke und rechte „Seiten“ des Schnittes bezeichnet.

Die Zahl $cc(G)$ eines Graphen G beschreibt die Anzahl der zusammenhängenden Komponenten („connected component“).

Für zwei Bags x, y eines Wurzelbaums gilt, dass y ein Nachkomme von x ist, falls es möglich ist ausgehend von y einen Weg zu x zu finden, der im Baum nur in Richtung des Wurzelknotens verläuft. Insbesondere ist x sein eigener Nachkomme.

Für zwei Integer a, b sagt die Gleichung $a \equiv b$ aus, dass a genau dann gerade ist, wenn auch b gerade ist. Zudem wird Iverson's Klammernotation verwendet. Falls p ein Prädikat ist, dann sei $[p]$ 1 (0) falls p wahr (unwahr) ist. Falls $\omega : U \rightarrow 1, \dots, N$, so bezeichnet $\omega(S) = \sum_{e \in S} \omega(e)$ für $S \subseteq U$.

Für eine Funktion s mit $s[v \rightarrow \alpha]$ schreiben wir die Funktion $s(v, s(v)) \cup (v, \alpha)$. Diese Definition funktioniert unabhängig davon, ob $s(v)$ bereits definiert wurde oder nicht.

Kapitel 2

(Nice) Tree Decomposition

2.1 Tree Decomposition

Definition 2.1. (Tree decomposition, [RS84]). Eine Tree Decomposition *eines (ungerichteten oder gerichteten) Graphen G* ist ein Baum \mathbb{T} in dem jedem Knoten $x \in \mathbb{T}$ eine Menge von Knoten $B_x \subseteq V$ (genannt „Bag“) zugeordnet ist, so dass

- für jede Kante $uv \in E$ existiert ein $x \in \mathbb{T}$, so dass $u, v \in B_x$
- falls $u \in B_x$ und $v \in B_y$, dann $v \in B_z$ für alle z auf dem Pfad von x nach y in \mathbb{T}

Das Konzept der Tree Decomposition wurde 1976 von Rudolf Halin[Hal76] eingeführt. Sie dient dazu die Baumweite zu definieren und Berechnungsprobleme auf Graphen schneller zu lösen.

Die Baumweite ist eine Zahl und beschreibt die „Baum-Ähnlichkeit“ eines Graphen. Die Baumweite $tw(\mathbb{T})$ einer Tree Decomposition \mathbb{T} ist die Größe des größten Bags minus eins. Die Baumweite eines Graphen G ist die minimale Baumweite aller möglichen Tree Decompositions von G .

Ein Beispiel für eine Tree Decomposition ist in Abbildung 2.1 gegeben. Der Ursprungsgraph ist in Abbildung 2.1.

2.2 Nice Tree Decomposition

Kloks[Klo94] führte die sogenannte Nice Tree Decomposition ein, welche oft für Algorithmen mit dynamischen Programmen genutzt werden. Da in Sektion 2.3 weitere Modifikationen auf Basis der Nice Tree Decomposition eingeführt werden, wird die hier vorgestellte Nice Tree Decomposition als *standardmäßige Nice Tree Decomposition* bezeichnet.

Definition 2.2. Eine standardmäßige Nice Tree Decomposition ist eine *Tree Decomposition*, für die gilt:

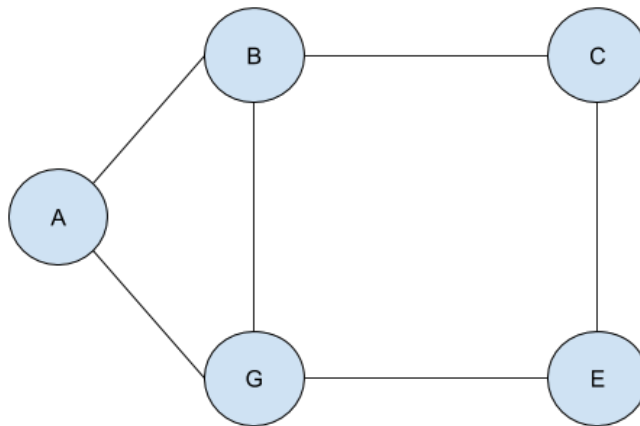


Abb. 2.1 Ursprungsgraph für eine Tree Decomposition

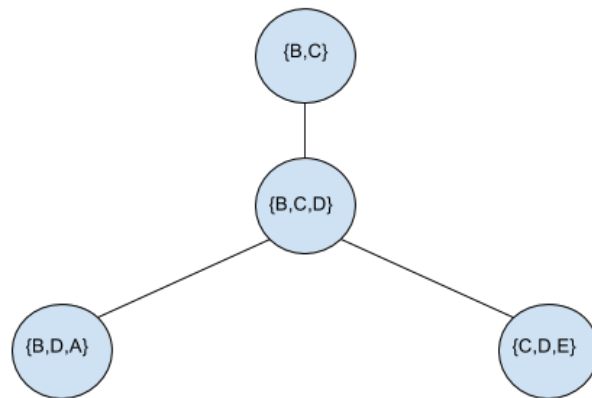


Abb. 2.2 Eine Tree Decomposition für den in Abbildung 2.1 gegebenen Ursprungsgraphen

- *jeder Bag besitzt höchstens zwei Kind-Knoten*
- *falls ein Bag zwei Kind-Knoten l, r besitzt, dann gilt $B_x = B_l = B_r$*
- *falls ein Bag x einen Kind-Knoten besitzt, dann gilt entweder $|B_x| = |B_y| + 1$ und $B_y \subseteq B_x$ oder $|B_x| + 1 = |B_y|$ und $B_x \subseteq B_y$*

2.3 Weitere Modifikationen

Für den von $[CNP^+]$ beschriebenen Algorithmus wird die standardmäßige Nice Tree Decomposition zusätzlich noch auf folgende Weise modifiziert:

Definition 2.3. (Nice Tree Decomposition). Eine Nice Tree Decomposition ist eine Tree Decomposition mit einem speziellen Bag z (Wurzel) mit $B_z = \emptyset$ und in der jeder Bag einer der folgenden Arten entspricht:

- **Leaf Bag:** ein Blatt x aus \mathbb{T} mit $B_x = \emptyset$.
- **Introduce Vertex Bag:** ein innerhalb von \mathbb{T} liegender Knoten x mit einem Kind-Knoten y für den gilt $B_x = B_y \cup \{v\}$ für einen $v \notin B_y$. Dieser Bag führt den Knoten v ein.
- **Introduce Edge Bag:** ein innerhalb von \mathbb{T} liegender Knoten x , der mit der Kante $uv \in E$ gekennzeichnet ist und einen Kind-Knoten y mit $u, v \in B_x = B_y$ besitzt. Dieser Knoten führt die Kante uv ein.
- **Forget Bag:** ein innerhalb von \mathbb{T} liegender Knoten x mit einem Kind-Knoten y , für den gilt $B_x = B_y \setminus \{v\}$ für ein $v \in B_y$. Dieser Bag vergisst den Knoten v .
- **Join Bag:** ein innerhalb von \mathbb{T} liegender Knoten x mit zwei Kind-Knoten l, r , für die $B_x = B_r = B_l$ gilt.

Zusätzlich wird gefordert, dass jede Kante aus E genau einmal eingeführt wird.

Ein Beispiel für eine Nice Tree Decomposition ist in Abbildung 2.3.

Sei eine Tree Decomposition gegeben, kann eine standardmäßige Nice Tree Decomposition von gleicher Breite in polynomieller Zeit gefunden werden [Klo94]. In der selber Laufzeit kann der Algorithmus für eine standardmäßige Nice Tree Decomposition modifiziert werden, so dass das Ergebnis zusätzlich die o.g. Kriterien erfüllt.

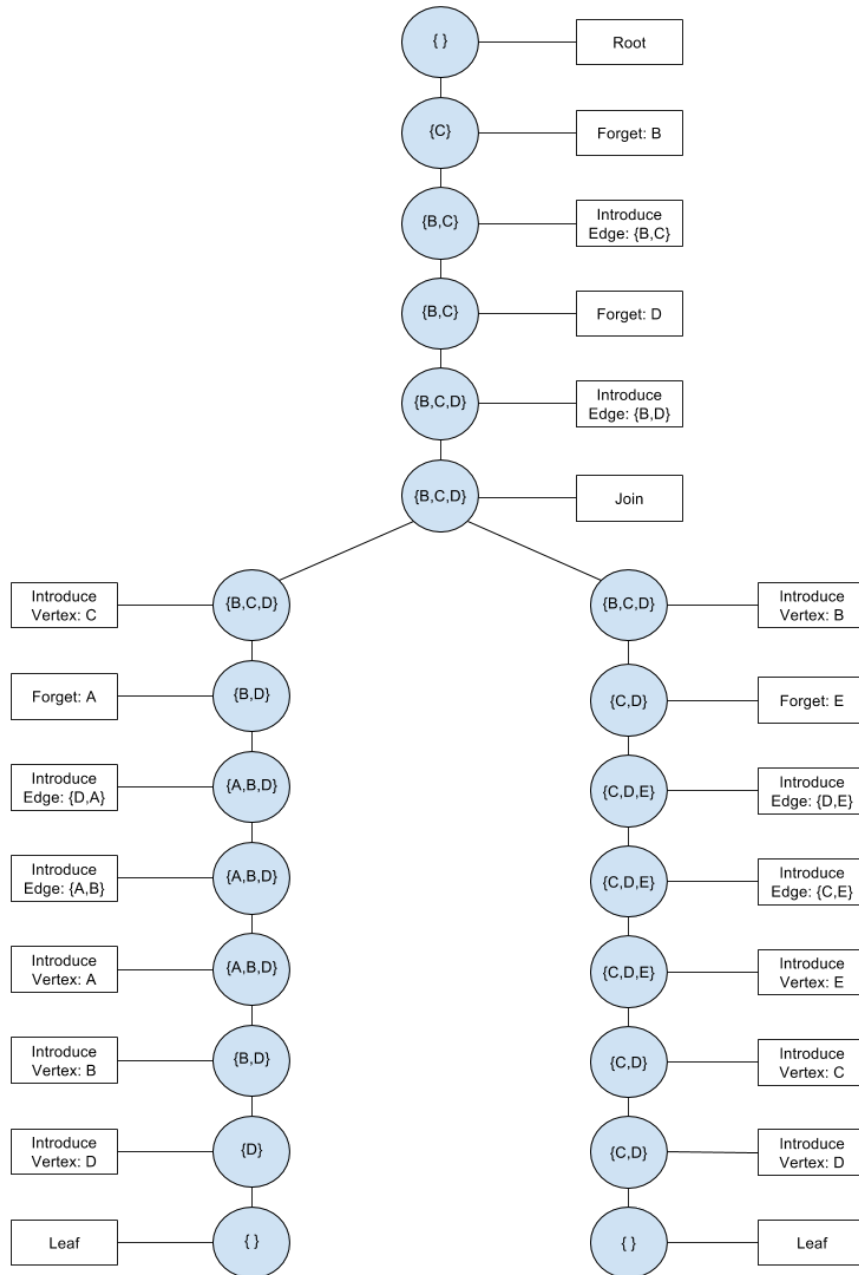


Abb. 2.3 Eine Nice Tree Decomposition für den Ursprungsgraphen in Abbildung 2.1

Kapitel 3

Cut & Count-Technik

3.1 Einführendes

Die Cut & Count-Technik wird benutzt um connectivity-type Probleme mithilfe von Randomisierung zu lösen. Dabei handelt es sich um Graphen-Probleme, bei denen zusammenhängende Submengen der Knoten gefunden werden müssen die problemspezifische Eigenschaften erfüllen müssen. Beispiele hierfür sind Longest Path, Steiner Tree, Feedback Vertex Set, uvm. Die Cut & Count-Technik verlangt zur Lösung der Probleme eine Randomisierung der Lösungsmenge, da sie nur korrekt funktioniert, wenn es eine oder keine Lösung gibt. Dies bedeutet, dass alle Lösungskandidaten eine zufällig und unabhängig gewählte Eigenschaft bekommen, welche zusätzlich Anforderungen erfüllen muss, um als gültige Lösungsmenge zu gelten. Diese Isolation der Lösungsmenge wird durch das Isolations-Lemma beschrieben.

3.2 Monte-Carlo-Algorithmus

Verwendet man die Cut & Count-Technik um ein Algorithmus zur Lösung der connectivity-type Probleme zu entwerfen, enthält man einen Monte-Carlo-Algorithmus. Dies bedeutet, dass der hervorgegangene Algorithmus niemals ein false-positiv¹ ausgeben kann aber zu einer gewissen Wahrscheinlichkeit ein false-negativ². Die Wahrscheinlichkeit eines false-negativ wird durch das Isolations-Lemma bestimmt. Da eine Ausgabe eine false-negativ einer bestimmten Wahrscheinlichkeit unterliegt ist es ratsam Monte-Carlo-Algorithmen mehrmals hintereinander auszuführen. Es begünstigt die Wahrscheinlichkeit eine Lösungsmenge zu finden.

¹ Der Algorithmus gibt eine positive Antwort auf das Problem, obwohl keine Lösung existiert

² Der Algorithmus gibt eine negative Antwort auf das Problem, obwohl eine Lösung existiert.

Der hervorgegangene Monte-Carlo-Algorithmus kann bei gegebener Tree-Decomposition mit Braumbreite t eines ungerichteten Graphens Lösungen bestimmen bei folgenden Probleme und Laufzeiten:

- Steiner Tree in $3^t |V|^{O(1)}$
- FeedBack Vertex Set in $3^t |V|^{O(1)}$
- ...

3.3 Isolation Lemma

Wie zuvor erwähnt ist die Cut & Count-Technik nur dann korrekt, wenn eine oder keine Lösung existiert. Da es bei vielen der connectivity-type Probleme sehr wahrscheinlich ist, dass mehrere Lösungen existieren, muss die Lösungsmenge reduziert werden. Dafür wird auf das Isolations-Lemma zurück gegriffen. Es ist wie folgt definiert:

A function $\omega : U \rightarrow \mathbb{Z}$ isolates a set family $\mathcal{F} \subseteq 2^U$ if there is a unique $S' \in \mathcal{F}$ with $\omega(S') = \min_{S \in \mathcal{F}} \omega(S)$

Im Paper wird es im Lemma 2.5 verwendet, um etwas über die Wahrscheinlichkeit der Isolation auszusagen:

Let $\mathcal{F} \subseteq 2^U$ be a set family over a universe U with $|\mathcal{F}| > 0$. For each $u \in U$, choose a weight $\omega(u) \in 1, 2, \dots, N$ uniformly and independently at random. Then

$$\text{prob}[\omega \text{ isolates } \mathcal{F}] \geq 1 - \frac{|U|}{N}$$

Durch die Wahl eines großen N kann somit die Wahrscheinlichkeit eines false-negativ stark reduziert werden aber es wird zugleich auch die Laufzeit des Algorithmus erhöht.

Kapitel 4

Cut & Count für Steiner Tree

4.1 Steiner Tree

Definition 4.1. Steiner Tree

Input: An undirected graph $G = (V, E)$, a set of terminals $T \subseteq V$ and an integer k .

Question: Is there a set $X \subseteq V$ of cardinality k such that $T \subseteq X$ and $G[X]$ is connected?

Bei dem Steiner-Tree Problem handelt es sich um ein connectivity-Typ Problem. In einen gegebenen Graphen sind einige Knoten als Terminale markiert. Nun wird nach einem Subgraphen innerhalb des Ursprungsgraphen gesucht, der alle Terminale enthält und aus k vielen Knoten insgesamt besteht. Desweiteren müssen alle Knoten innerhalb des Subgraphen über Kanten untereinander erreichbar sein.

4.2 Cut

Zu Beginn des Cut-Parts wird eine zufällige Gewichtsfunktion $\omega : V \rightarrow \{1, \dots, N\}$ definiert. Diese wird für die Isolation der Lösungsmenge verwendet. Desweiteren wird im Cut-Part die Menge \mathcal{R}_W definiert. Sie ist die Menge aller Teilmengen von X aus V mit $T \subseteq X$, $\omega(X) = W$ und $|X| = k$. Somit stellt die Menge \mathcal{R}_W die Menge aller Lösungskandidaten dar. Eine weitere zu definierende Menge ist $\mathcal{S}_W = \{X \in \mathcal{R}_W \mid G[X] \text{ ist zusammenhängend}\}$. \mathcal{S}_W . Somit bildet die Menge \mathcal{S}_W Lösungen für ein gegebenes bestimmtes Gewicht W . $\cup_W \mathcal{S}_W$ bildet so die Lösungsmenge. Gibt es nun ein Gewicht W für das die Menge nicht leer ist, so gibt der Algorithmus eine positive Antwort. Von der Menge der Terminalknoten wird ein Terminal als v_1 -Terminal festgelegt. Dieses dient dazu, dass bei der Bildung von konsistenten Cuts keine Cuts doppelt gezählt werden. Die konsistenten Cuts werden in der Menge \mathcal{C}_W beschrieben. Diese bilden die Menge aller Subgraphen, die einen konsistenten Cut $(X, (X_1, X_2))$ bilden, wobei $X \in \mathcal{R}_W$ und $v_1 \in X_1$. Die Anzahl der konsistenten Cuts sind im Lemma 3.3 des Papers beschrieben:

Let $G = (V, E)$ be a graph and let X be a subset of vertices such that $v_1 \in X \subseteq V$. The number of consistently cut subgraphs $(X, (X_1, X_2))$ such that $v_1 \in X_1$ is equal to $2^{cc(G[X])-1}$.

4.3 Count

Aus Lemma 3.3 ist bekannt: $|\mathcal{C}| = \sum_{X \in \mathcal{R}} 2^{cc(G[X])-1}$. Wir legen W fest und ignorieren die Indices: $|\mathcal{C}| \equiv |\{X \in \mathcal{R} | cc(G[X]) = 1\}| = |\mathcal{S}|$. In Worte gefasst bedeutet dies, dass die Anzahl der konsistenten Cuts eines Graphen modulo zwei gleich die Anzahl der Lösungen ist. Im Lemma 3.4 trifft das Paper dazu eine Aussage: Let G, ω, \mathcal{C}_W and \mathcal{S}_W be as defined above. Then for every W , $|\mathcal{S}_W| \equiv |\mathcal{C}_W| \cdot |\mathcal{C}_W|$ modulo 2 kann mit dynamischen Programm auf der NTD \mathbb{T} berechnet werden. für jeden Bag $x \in \mathbb{T}$, integers $0 \leq i \leq k, 0 \leq w \leq kN$ und Färbung $s \in \{0, 1_1, 1_2\}^{B_x}$ definiere:

- $\mathcal{R}_x(i, w) = \{X \subseteq V_x | (T \cap V_x) \subseteq X \wedge |X| = i \wedge \omega(X) = w\}$
- $\mathcal{C}_x(i, w) = \{(X, (X_1, X_2)) | X \in \mathcal{R}_x(i, w) \wedge (X, (X_1, X_2)) \text{ is a consistently cut subgraph of } G_x \wedge (v_1 \in V_x \Rightarrow v_1 \in X_1)\}$
- $\mathcal{A}_x(i, w, s) = |\{(X, (X_1, X_2)) \in \mathcal{C}_x(i, w) | (s(v) = 1_j \Rightarrow v \in X_j) \wedge (s(v) = 0 \Rightarrow v \notin X)\}|$

Die Färbungen der Knoten gibt an ob sie zu einem konsistenten Cut gehören und wenn ja zu welcher der beiden Teilmengen des konsistenten Cuts sie gehören. Färbung $s \in \{0, 1_1, 1_2\}^{B_x}$ der Knoten aus B_x bzgl. der Menge C_x

- $s[v] = 0 \Rightarrow v \notin X$
- $s[v] = 1_1 \Rightarrow v \in X_1$
- $s[v] = 1_2 \Rightarrow v \in X_2$

$A_x(i, w, s)$ zählt alle möglichkeiten die Knoten gemäß der Definition zu färben. Ob der Algorithmus eine Lösung gefunden hat kann im Wurzel-Knoten eingesehen werden.

4.4 Dynamisches Programm

4.5 Monte-Carlo Algorithmus und Laufzeit

Kapitel 5

Implementierung

5.1 Nice Tree Decomposition

Es wurde ein Algorithmus entwickelt, der als Eingabe eine standardmäßige Nice Tree Decomposition \mathbb{T} eines Graphen G erhält und eine Nice Tree Decomposition (siehe Kapitel 5.1) ausgibt. Da der Fokus dieser Arbeit auf der Implementierung des dynamischen Programms des Cut & Count-Algorithmus liegt, wurde der Algorithmus nicht hinsichtlich der in [Klo94] beschriebenen polynomiellen Laufzeit optimiert.

Der Algorithmus iteriert mehrmals in symmetrischer Reihenfolge (ausgehend von der Wurzel linksseitig absteigend) über \mathbb{T} und fügt dabei die fehlenden Knoten ein. Zu Beginn werden am bisherigen Wurzelknoten so lange „Forget“-Knoten angehängt bis noch ein Knoten des Ursprungsgraphen im Bag verbleibt. Anschließend wird ein letzter Knoten mit leerem Bag als neuer Wurzelknoten hinzugefügt. Ähnlich wird hinsichtlich der Blattknoten verfahren. Entsprechend der Differenz eines leeren Bags und der Bags der bisherigen Blattknoten werden neue „Introduce-Vertex“-Knoten eingefügt und am Ende jedes Pfades ein Knoten mit leerem Bag als neuer Blattknoten angehängt. Für bestehende „Join“-Knoten werden die Bags der beiden Kindknoten verglichen. Sofern sie nicht denselben Bag wie der „Join“-Knoten haben, werden neue Knoten („Forget“, „Introduce Vertex“) eingefügt, bis die Bags identisch mit dem Elternknoten sind. Anschließend wird über den Graphen iteriert. Zuletzt wird für jede Kante e des Ursprungsgraphen G über den Graphen iteriert. Beim ersten gemeinsamen Auftreten der Knoten der Kante e innerhalb eines Bags, wird oberhalb des aktuellen Knoten im Graphen ein neuer „Introduce Edge“-Knoten eingeführt und mit den Knoten der Kante e gekennzeichnet.

5.2 Dynamisches Programm

5.3 Evaluierung

5.4 Ausblick

Kapitel 6

Zusammenfassung

6.1 Text

Use the template *chapter.tex* together with the Springer document class SVMono (monograph-type books) or SVMult (edited books) to style the various elements of your chapter content in the Springer layout.

Kapitel 7

Chapter Heading

7.1 Section Heading

Use the template *chapter.tex* together with the Springer document class SVMono (monograph-type books) or SVMult (edited books) to style the various elements of your chapter content in the Springer layout.

7.2 Section Heading

Instead of simply listing headings of different levels we recommend to let every heading be followed by at least a short passage of text. Furtheron please use the L^AT_EX automatism for all your cross-references and citations.

Please note that the first line of text that follows a heading is not indented, whereas the first lines of all subsequent paragraphs are.

Use the standard `equation` environment to typeset your equations, e.g.

$$a \times b = c, \tag{7.1}$$

however, for multiline equations we recommend to use the `eqnarray` environment¹.

$$\begin{array}{l} a \times b = c \\ \mathbf{a} \cdot \mathbf{b} = \mathbf{c} \end{array} \tag{7.2}$$

¹ In physics texts please activate the class option `vecphys` to depict your vectors in *boldface-italic* type - as is customary for a wide range of physical subjects.

7.2.1 Subsection Heading

Instead of simply listing headings of different levels we recommend to let every heading be followed by at least a short passage of text. Furtheron please use the \LaTeX automatism for all your cross-references and citations as has already been described in Sect. 7.2.

Please do not use quotation marks when quoting texts! Simply use the `quotation` environment – it will automatically render Springer’s preferred layout.

7.2.1.1 Subsubsection Heading

Instead of simply listing headings of different levels we recommend to let every heading be followed by at least a short passage of text. Furtheron please use the \LaTeX automatism for all your cross-references and citations as has already been described in Sect. 7.2.1, see also Fig. 7.1²

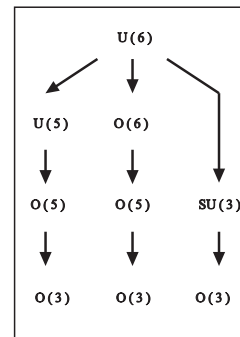
Please note that the first line of text that follows a heading is not indented, whereas the first lines of all subsequent paragraphs are.

Paragraph Heading

Instead of simply listing headings of different levels we recommend to let every heading be followed by at least a short passage of text. Furtheron please use the \LaTeX automatism for all your cross-references and citations as has already been described in Sect. 7.2.

Please note that the first line of text that follows a heading is not indented, whereas the first lines of all subsequent paragraphs are.

Abb. 7.1 If the width of the figure is less than 7.8 cm use the `sidecaption` command to flush the caption on the left side of the page. If the figure is positioned at the top of the page, align the sidecaption with the top of the figure – to achieve this you simply need to use the optional argument `[t]` with the `sidecaption` command



² If you copy text passages, figures, or tables from other works, you must obtain *permission* from the copyright holder (usually the original publisher). Please enclose the signed permission with the manuscript. The sources must be acknowledged either in the captions, as footnotes or in a separate section of the book.

For typesetting numbered lists we recommend to use the `enumerate` environment – it will automatically render Springer’s preferred layout.

1. Livelihood and survival mobility are oftentimes coutcomes of uneven socioeconomic development.
 - a. Livelihood and survival mobility are oftentimes coutcomes of uneven socioeconomic development.
 - b. Livelihood and survival mobility are oftentimes coutcomes of uneven socioeconomic development.
2. Livelihood and survival mobility are oftentimes coutcomes of uneven socioeconomic development.

Subparagraph Heading

In order to avoid simply listing headings of different levels we recommend to let every heading be followed by at least a short passage of text. Use the \LaTeX automatism for all your cross-references and citations as has already been described in Sect. 7.2, see also Fig. 7.2.

Please note that the first line of text that follows a heading is not indented, whereas the first lines of all subsequent paragraphs are.

For unnumbered list we recommend to use the `itemize` environment – it will automatically render Springer’s preferred layout.

- Livelihood and survival mobility are oftentimes coutcomes of uneven socioeconomic development, cf. Table 7.1.
 - Livelihood and survival mobility are oftentimes coutcomes of uneven socioeconomic development.
 - Livelihood and survival mobility are oftentimes coutcomes of uneven socioeconomic development.
- Livelihood and survival mobility are oftentimes coutcomes of uneven socioeconomic development.

Run-in Heading Boldface Version Use the \LaTeX automatism for all your cross-references and citations as has already been described in Sect. 7.2.

Run-in Heading Italic Version Use the \LaTeX automatism for all your cross-references and citations as has already been described in Sect. 7.2.

7.3 Section Heading

Instead of simply listing headings of different levels we recommend to let every heading be followed by at least a short passage of text. Furtheron please use the

Abb. 7.2 Please write your figure caption here

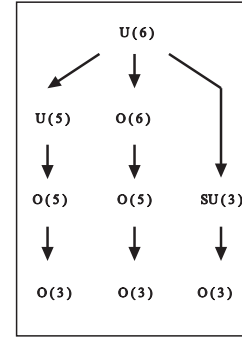


Tabelle 7.1 Please write your table caption here

Classes	Subclass	Length	Action Mechanism
Translation	mRNA ^a	22 (19–25)	Translation repression, mRNA cleavage
Translation	mRNA cleavage	21	mRNA cleavage
Translation	mRNA	21–22	mRNA cleavage
Translation	mRNA	24–26	Histone and DNA Modification

^a Table foot note (with superscript)

\LaTeX automatism for all your cross-references and citations as has already been described in Sect. 7.2.

Please note that the first line of text that follows a heading is not indented, whereas the first lines of all subsequent paragraphs are.

If you want to list definitions or the like we recommend to use the Springer-enhanced `description` environment – it will automatically render Springer’s preferred layout.

- Type 1 That addresses central themes pertaining to migration, health, and disease. In Sect. 7.1, Wilson discusses the role of human migration in infectious disease distributions and patterns.
- Type 2 That addresses central themes pertaining to migration, health, and disease. In Sect. 7.2.1, Wilson discusses the role of human migration in infectious disease distributions and patterns.

7.3.1 Subsection Heading

In order to avoid simply listing headings of different levels we recommend to let every heading be followed by at least a short passage of text. Use the \LaTeX automatism for all your cross-references and citations as has already been described in Sect. 7.2.

Please note that the first line of text that follows a heading is not indented, whereas the first lines of all subsequent paragraphs are.

If you want to emphasize complete paragraphs of texts we recommend to use the newly defined Springer class option `graybox` and the newly defined environment `svgraybox`. This will produce a 15 percent screened box 'behind' your text.

If you want to emphasize complete paragraphs of texts we recommend to use the newly defined Springer class option and environment `svgraybox`. This will produce a 15 percent screened box 'behind' your text.

7.3.1.1 Subsubsection Heading

Instead of simply listing headings of different levels we recommend to let every heading be followed by at least a short passage of text. Furtheron please use the \LaTeX automatism for all your cross-references and citations as has already been described in Sect. 7.2.

Please note that the first line of text that follows a heading is not indented, whereas the first lines of all subsequent paragraphs are.

Theorem 7.1. *Theorem text goes here.*

Definition 7.1. Definition text goes here.

Beweis. Proof text goes here. \square

Paragraph Heading

Instead of simply listing headings of different levels we recommend to let every heading be followed by at least a short passage of text. Furtheron please use the \LaTeX automatism for all your cross-references and citations as has already been described in Sect. 7.2.

Note that the first line of text that follows a heading is not indented, whereas the first lines of all subsequent paragraphs are.

Theorem 7.2. *Theorem text goes here.*

Definition 7.2. Definition text goes here.

Beweis. Proof text goes here. \square

Danksagung If you want to include acknowledgments of assistance and the like at the end of an individual chapter please use the `acknowledgement` environment – it will automatically render Springer's preferred layout.

Appendix

When placed at the end of a chapter or contribution (as opposed to at the end of the book), the numbering of tables, figures, and equations in the appendix section continues on from that in the main text. Hence please *do not* use the `appendix` command when writing an appendix at the end of your chapter or contribution. If there is only one the appendix is designated “Appendix”, or “Appendix 1”, or “Appendix 2”, etc. if there is more than one.

$$a \times b = c \tag{7.3}$$

Problems

7.1. A given problem or Exercise is described here. The problem is described here. The problem is described here.

7.2. Problem Heading

- (a) The first part of the problem is described here.
- (b) The second part of the problem is described here.

Sachverzeichnis

citations, 14
cross-references, 14

paragraph, 15
permission to print, 14

Literaturverzeichnis

- [CNP⁺] Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michal Pilipczuk, Joham MM van Rooij, and Jakub Onufry Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 150–159. IEEE.
- [Hal76] Rudolf Halin. S-functions for graphs. *Journal of Geometry*, 8(1):171–186, 1976.
- [Klo94] Ton Kloks. *Treewidth: computations and approximations*, volume 842. Springer Science & Business Media, 1994.
- [RS84] Neil Robertson and Paul D Seymour. Graph minors. iii. planar tree-width. *Journal of Combinatorial Theory, Series B*, 36(1):49–64, 1984.