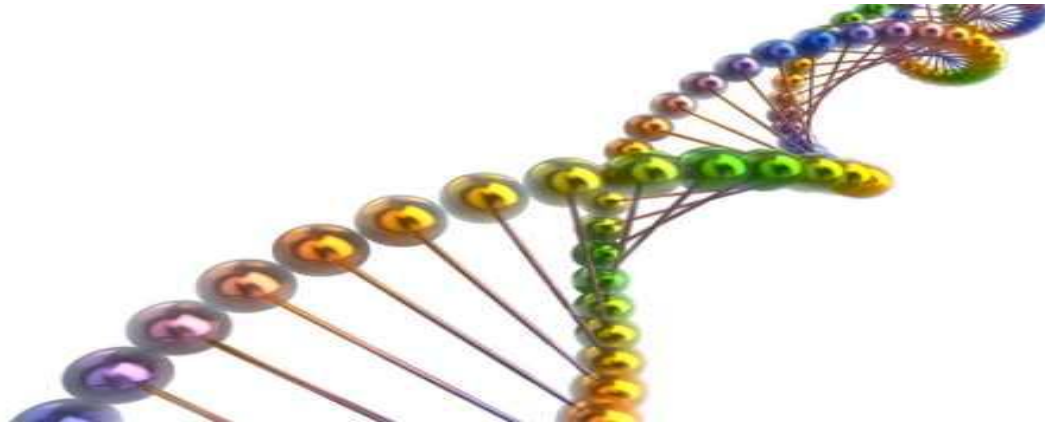




3. Konzepte des DNA Computing

**Naturwissenschaftlich motivierte
formale Modelle**

Henning Bordihn

GeneChips

- Ab der *Intrepid*-Klasse: Bionukleare Gelpacks
- *Realität?:*



- Tools zur **Genanalyse**, bestimmen Abweichungen im Erbgut, z.B.
 - gesunde vs. kranke Zelle,
 - Wildtyp vs. Mutant, ...
- realisieren sog. Microarrays
- tragen Millionen Polymere (DNA-Vergleichssequenzen)

Was ist DNA Computing?

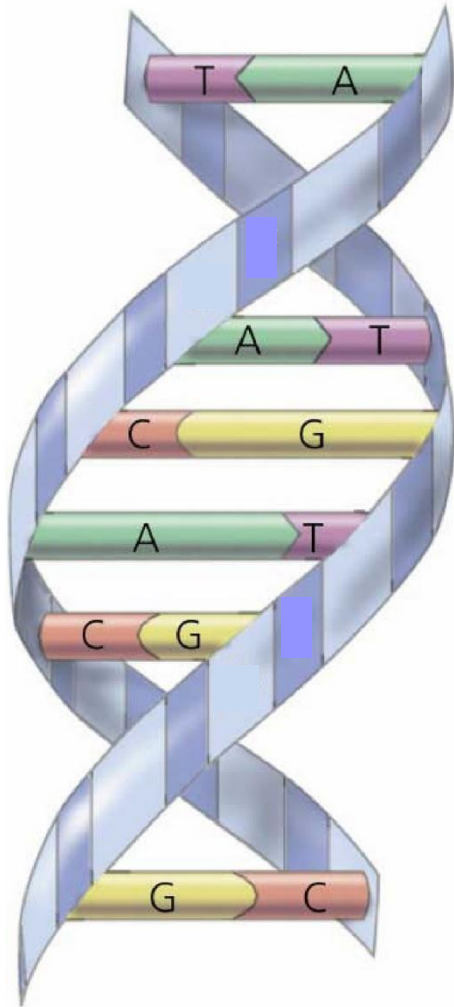
- *Computing*: algorithmische Überführung von Eingabe- in Ausgabedaten
- *DNA-Computing*:
 - Eingabe- und Ausgabedaten werden in DNA-Molekülketten kodiert
 - algorithmische Schritte sind Operationen, die DNA-Molekülketten manipulieren
- Miniaturisierung durch Übergang auf die molekulare Ebene
 - DNA-Computing
 - Quantum-Computing

Facetten des DNA Computing

- *in vivo*: DNA-Operationen in der lebenden Zelle
 - Erkennen von Zusammenhängen
- *in vitro*: DNA-Operationen außerhalb der lebenden Zelle
 - Nachahmung der *in-vivo*-Operationen unter Laborbedingungen
- *in razione*: mathematische Konzepte
 - durch DNA-Operationen inspiriert
- *in silico*: Realisierung solcher Konzepte mit herkömmlichen Methoden der Informatik

3.1 Struktur und grundlegende Eigenschaften von DNA

DNA-Stränge



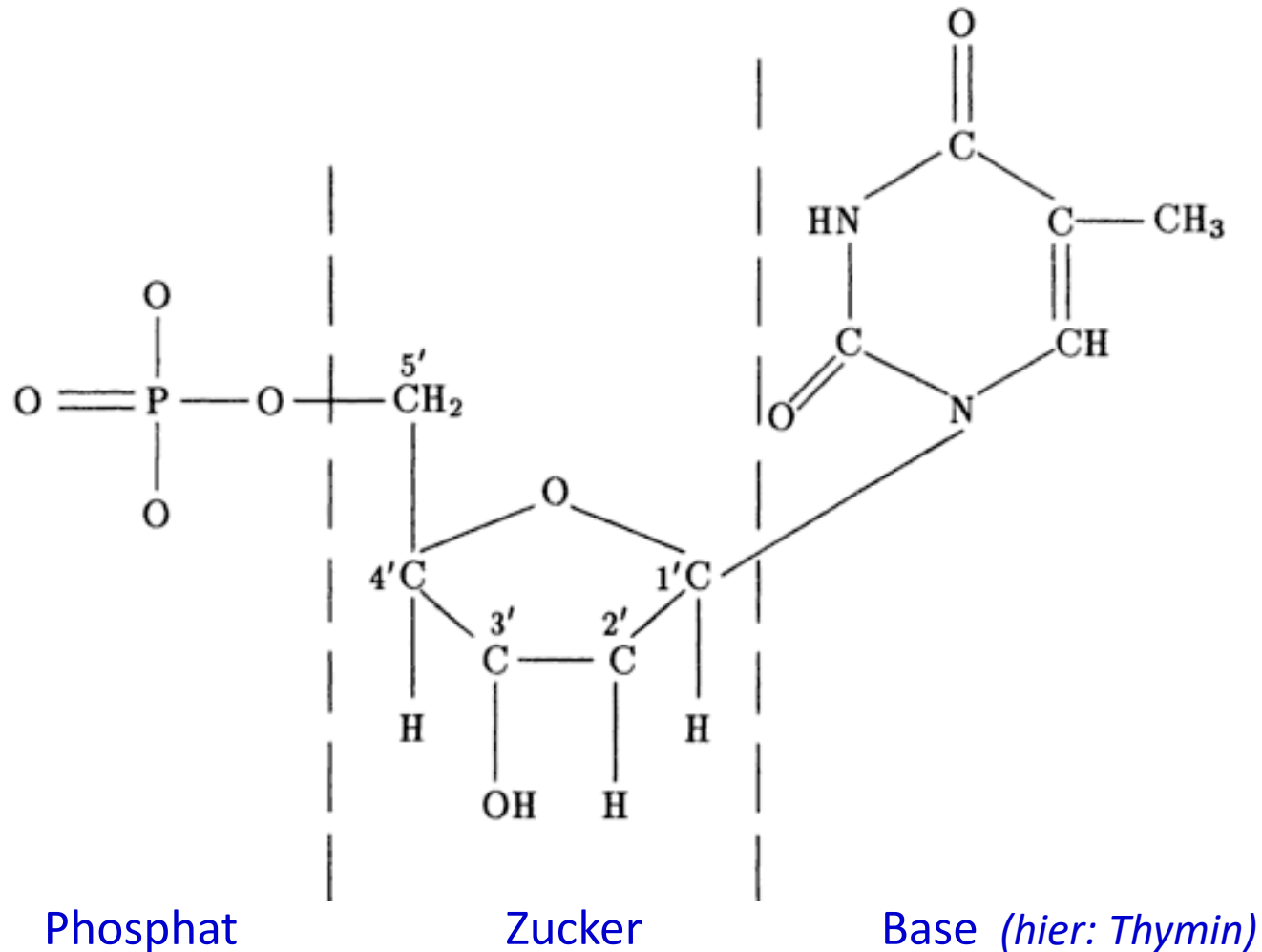
gewundene Doppelhelix,
bestehend aus:

- Adenin- (**A**),
- Cytosin- (**C**),
- Guanin- (**G**),
- Thymin- (**T**)

Nukleotiden

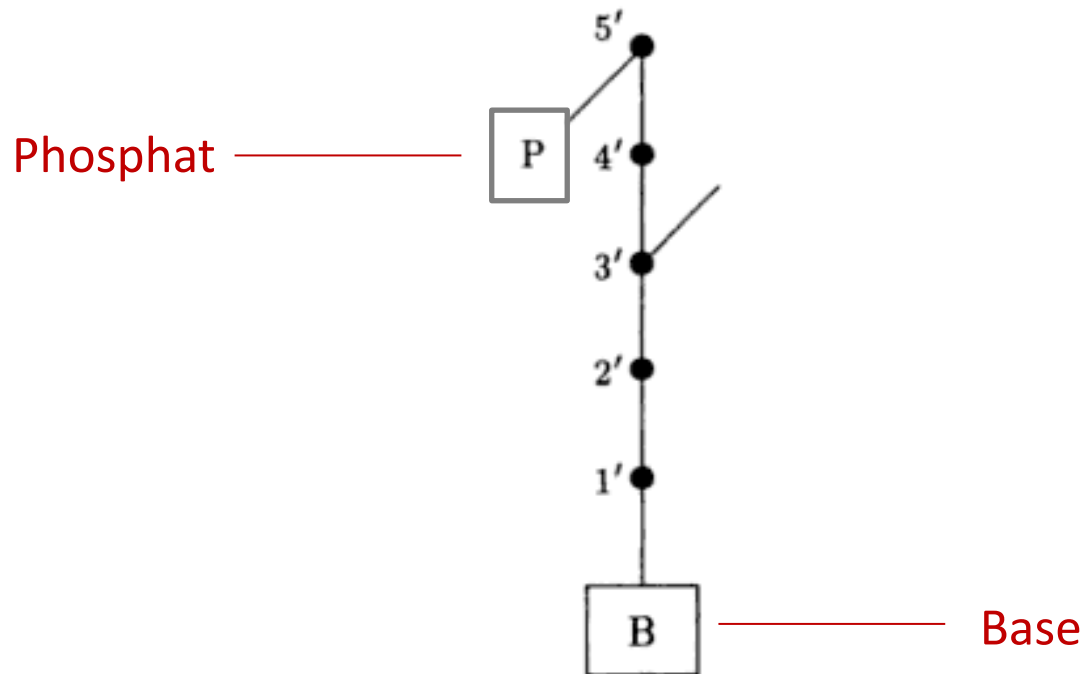
Chemische Struktur eines Nukleotids

[PRS 98]



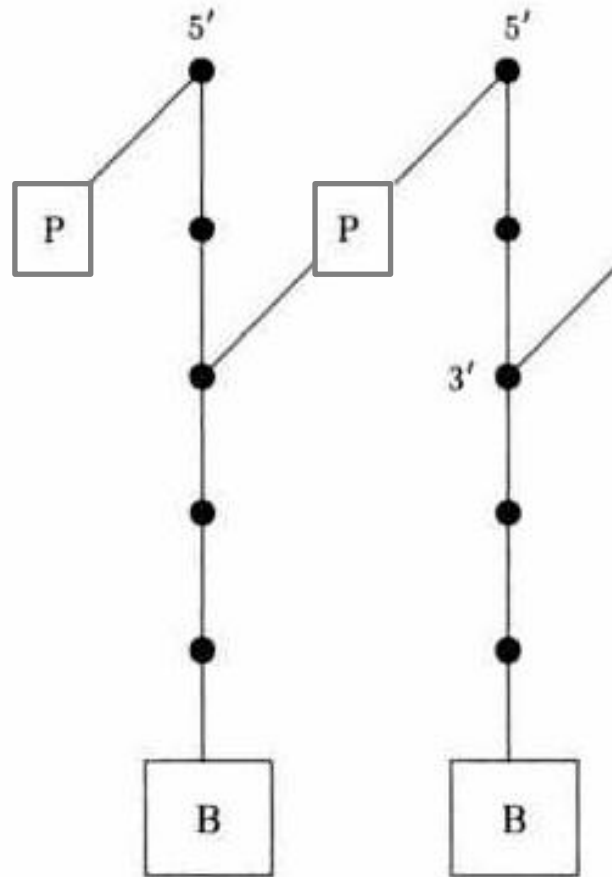
Schematische Darstellung eines Nukleotids

[PRS 98]



Phosphodiesterbindung

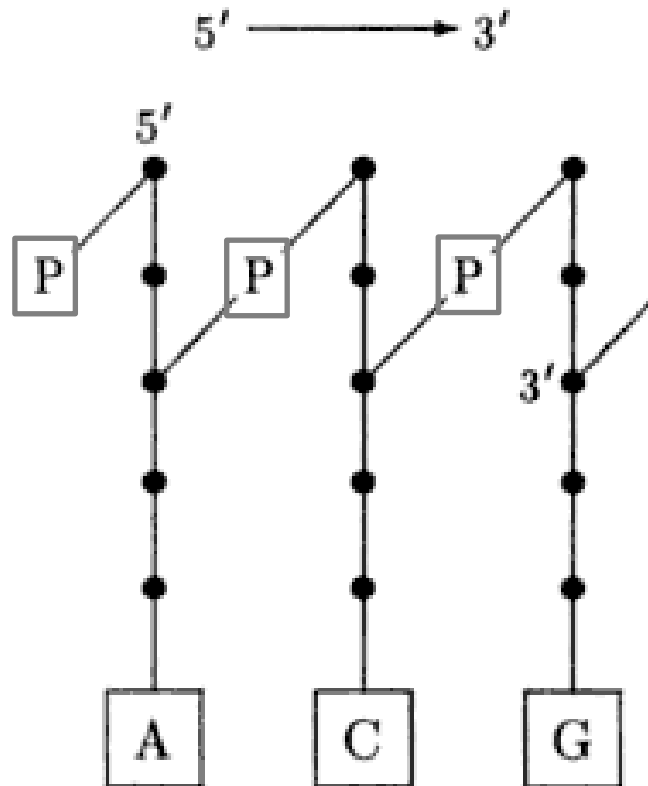
[PRS 98]



chemisch starke
Bindung zwischen
zwei Nukleotiden
zwischen der
Phosphorgruppe
an 5' und der
OH-Gruppe an 3'

Phosphodiesterbindung

[PRS 98]



—— Leserichtung

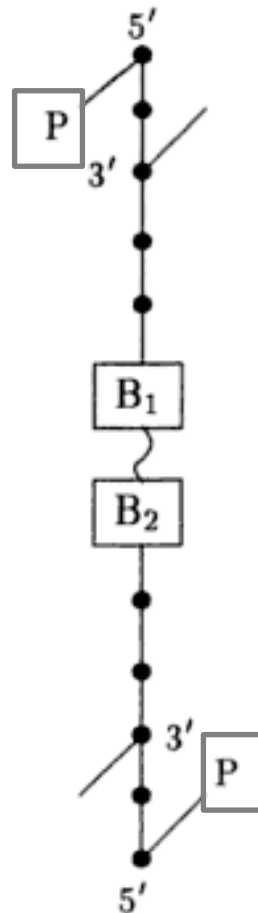
Repräsentation eines
DNA-Einzelstrangs
als String:

5'ACG3'

—— Nukleotidsequenz

Wasserstoffbindung

[PRS 98]



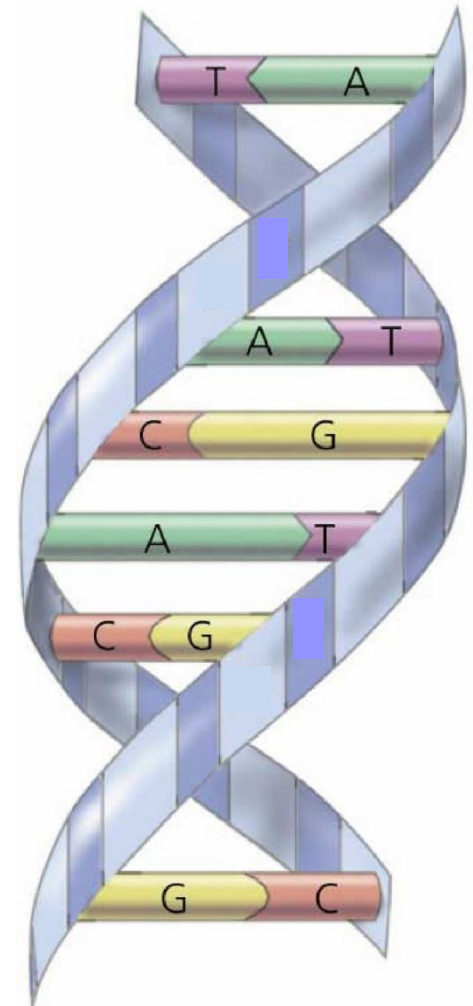
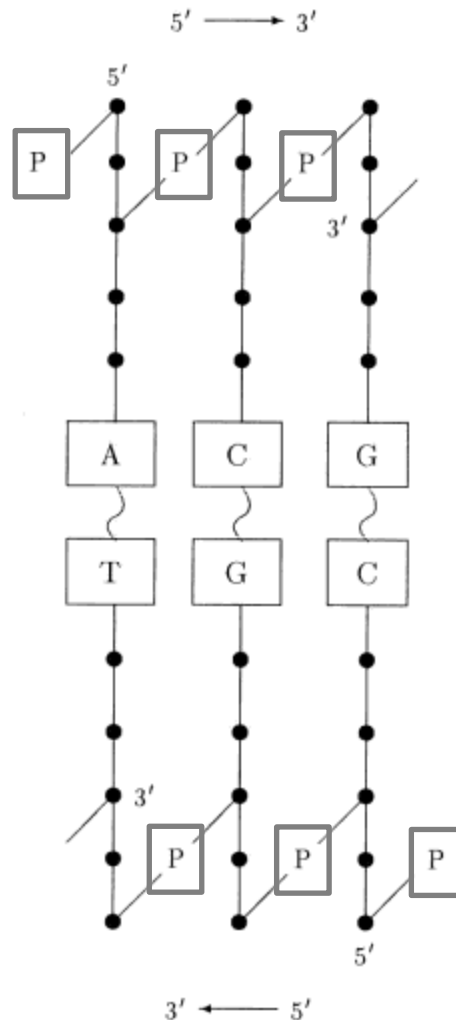
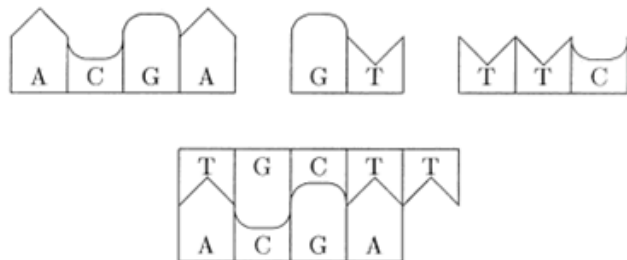
chemisch schwache
Bindung zwischen
zwei Nukleotiden

Watson-Crick-
Komplementarität:

entweder $B_1 = A$ und $B_2 = T$
oder $B_1 = C$ und $B_2 = G$

Bildung eines Doppelstrangs

[PRS 98]

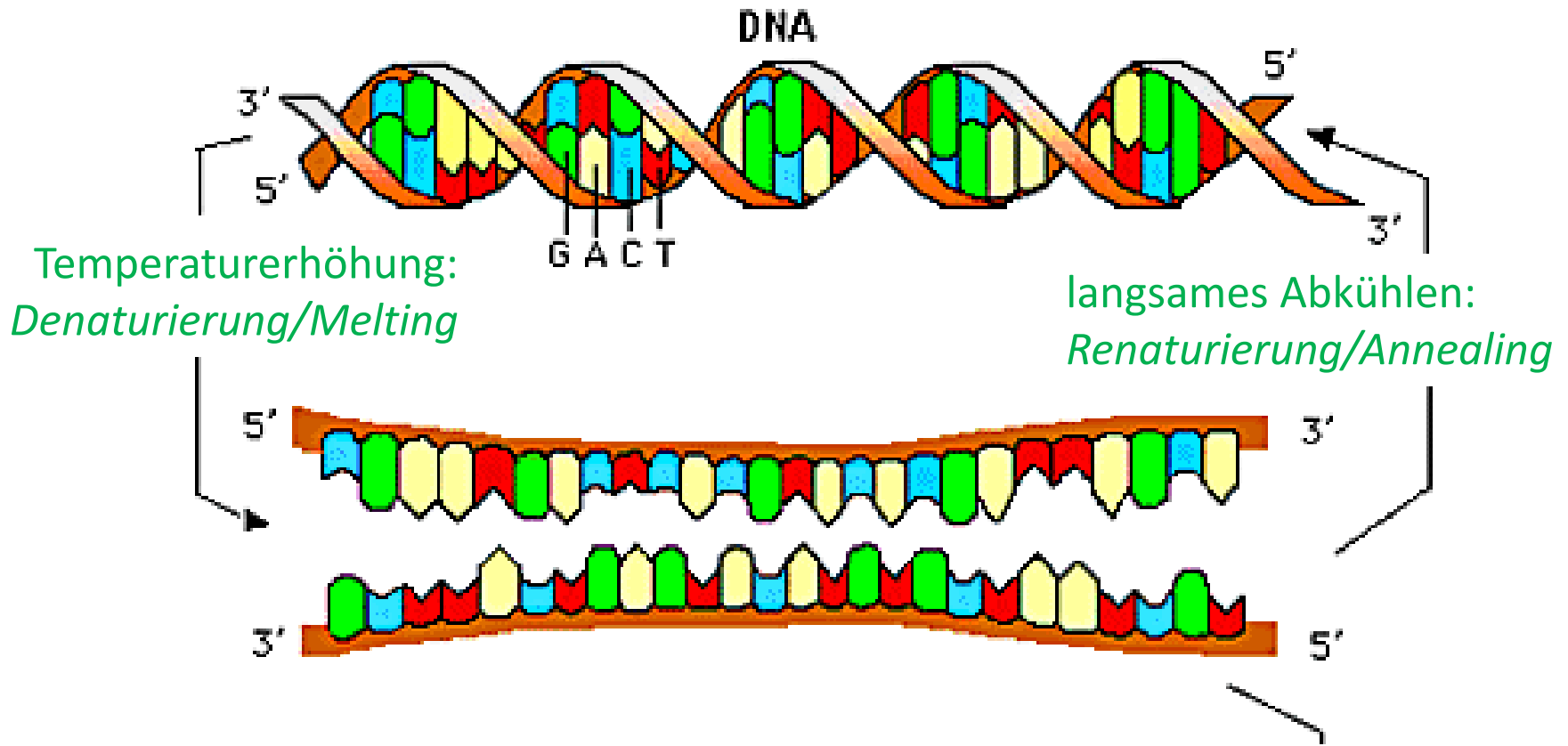


3.2 Einige DNA-Operationen

DNA-Replikation (*in vivo*)

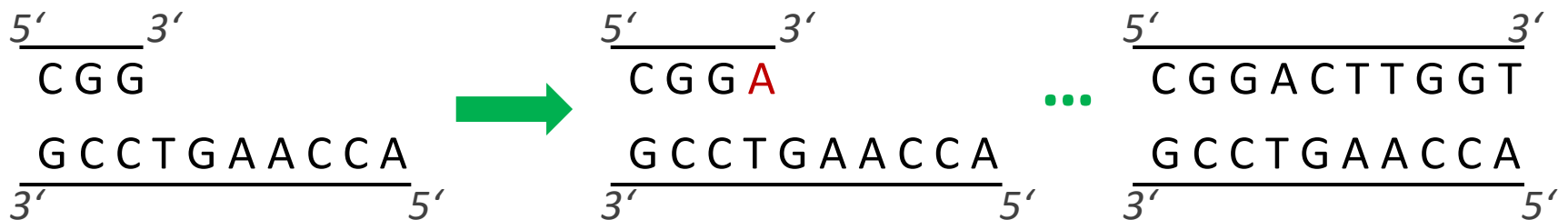
- Trennung des DNA-Doppelstrangs in Einzelstränge durch Lösen der Wasserstoff-Bindungen
- Ergänzen der Einzelstränge auf Basis der Watson-Crick-Komplementarität
- Operationen können *in vitro* nachgeahmt werden

Denaturieren - Renaturieren



DNA-Kopie

- Voraussetzungen:
 1. ein DNA-Einzelstrang (*Template*), nach Denaturierung
 2. ein am 3'-Ende des Templates bereits gebundenes Anfangsstück (*Primer*)
 3. genügend viele Nukleotide in der Lösung
 4. Enzym *Polymerase*
(katalysiert Anbindung von Nukleotiden am freien 3'-Ende)

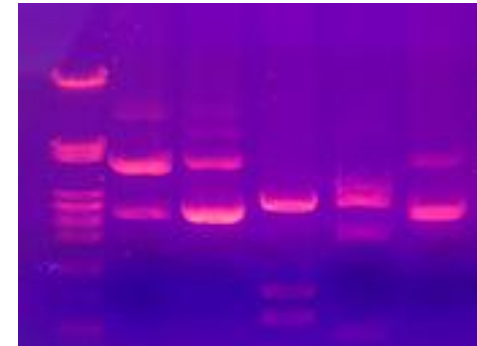
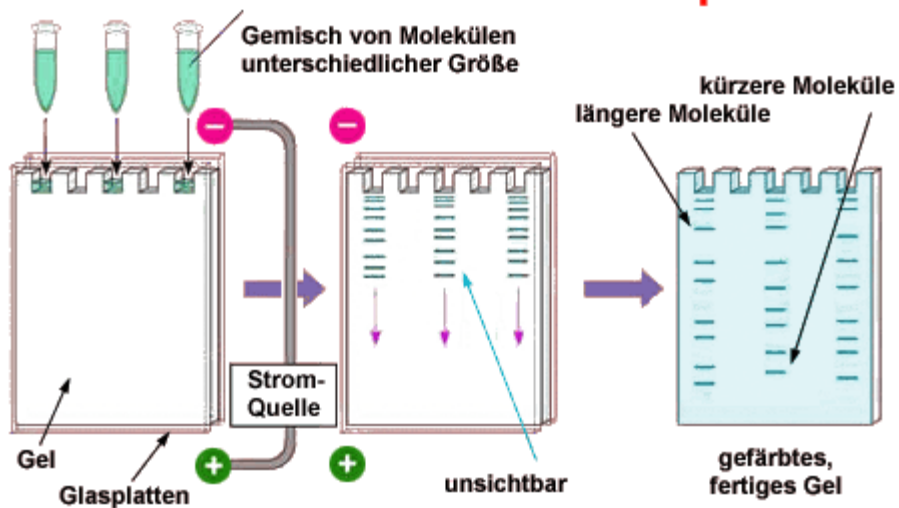


Filtern von DNA-Strängen

- Lösung S mit DNA-Polymeren (Einzelsträngen)
- Ziel: Ausfiltern (Entfernen) von allen Polymeren α
- Methode: (stark vereinfacht)
 1. Filter mit Komplementär-Polymeren $\bar{\alpha}$ präparieren
 2. S durch Filter geben + *Annealing*
- Ergebnis:
 1. Lösung S' (ohne α)
 2. Polymere α (nach *Denaturieren* vom Filter)
- verfeinerte Varianten existieren

Gel-Elektrophorese

Gelelektrophorese



<http://www.egbeck.de/skripten/13/bs13-11.htm>

3.3 Experimente *in vitro*

Prinzipielles Vorgehen *in vitro*

- **gegeben:** Instanz eines Berechnungsproblems
- Kodierung der Instanz mit Hilfe von DNA-Sequenzen
- Lösung des Problems durch geeignete Folge von DNA-Operationen (*DNA-Algorithmus*)
- Auslesen der Problemlösung
 - Frage nach der Existenz einer bestimmten DNA-Sequenz
- besonders interessant für Probleme mit sehr hoher Komplexität

Das Hamilton-Pfad-Problem

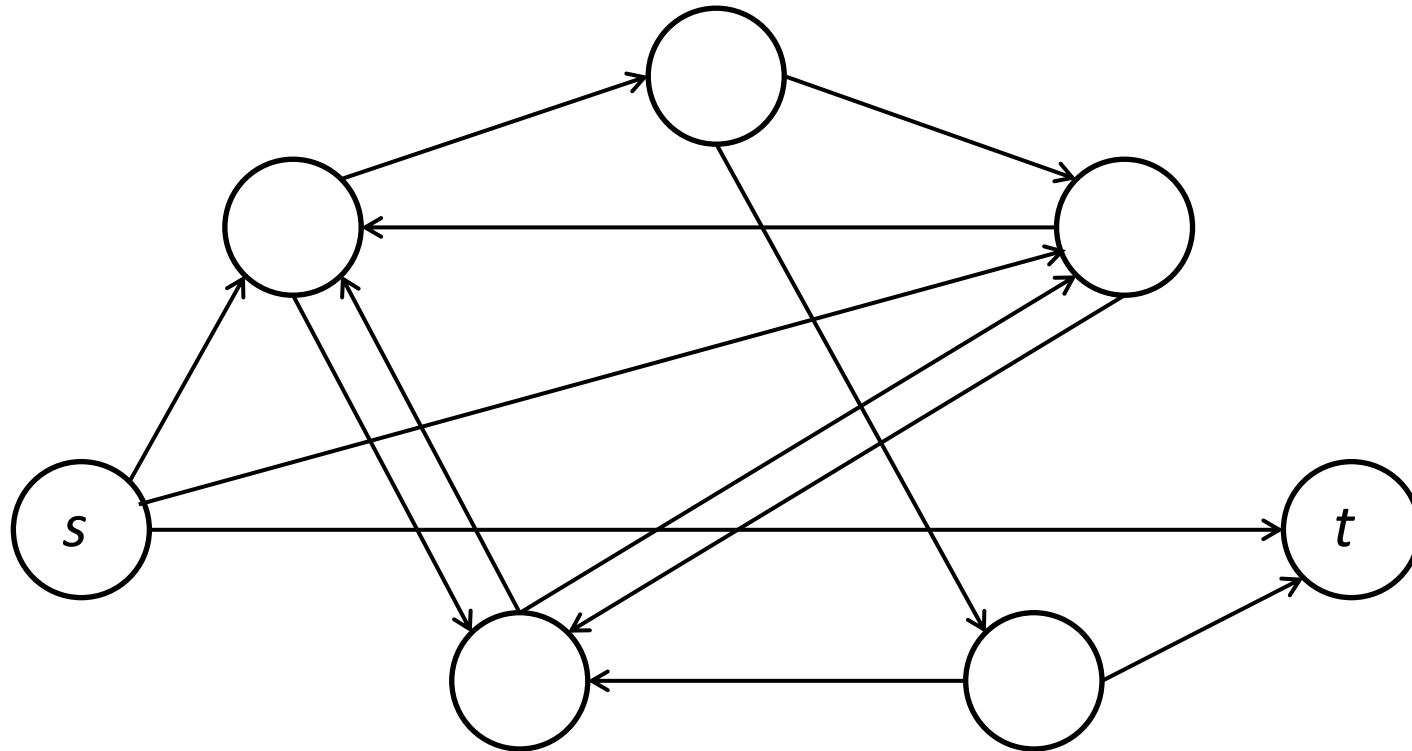
Gegeben: gerichteter Graph G und zwei seiner Knoten s und t

Frage: Gibt es einen Pfad von s nach t ,
der jeden Knoten von G genau einmal enthält?

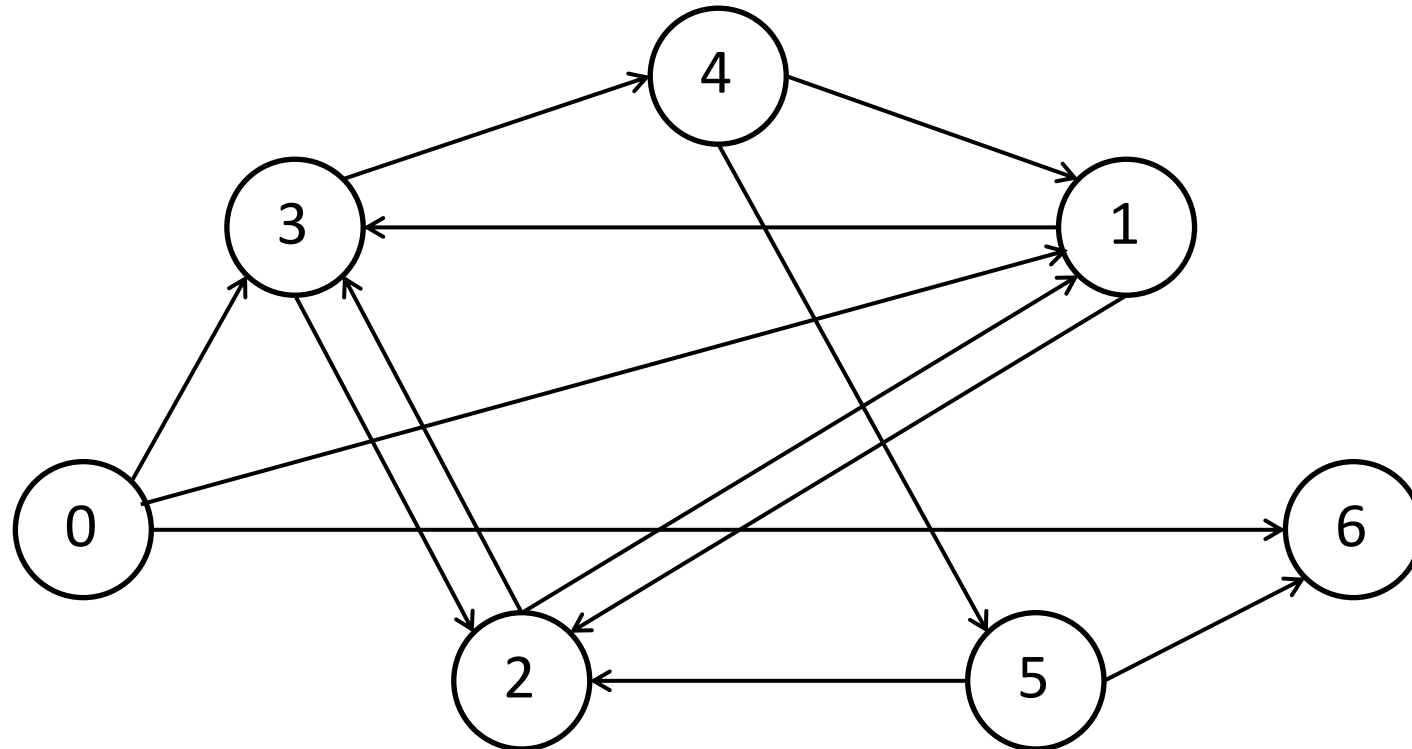
Das Hamilton-Pfad-Problem ist **NP-vollständig**!

- Alle bekannten (herkömmlichen) Algorithmen basieren auf *vollständigem Durchmustern* aller Lösungsmöglichkeiten (exponentielle Laufzeit).
- Es ist unwahrscheinlich, dass effizientere Algorithmen gefunden werden.

Der Graph in Adleman's Experiment



Der Graph in Adlemans Experiment



Hamilton-Pfad: 0, 1, 2, 3, 4, 5, 6

Adlemans Lösung

Sei G gerichteter Graph mit n Knoten, darunter s und t .

1. Erzeugen von Kodierungen zufällig gewählter Pfade in G in sehr großer Anzahl.
2. Massives Vermehren von Pfaden, die in s beginnen oder t enden (*Polymerase-Kettenreaktion mit s bzw. $\bar{t}$ als Primer*). Ausfiltern aller Pfade, die nicht in s beginnen *und* t enden.
3. Filtere alle Pfade aus, die nicht genau n Knoten enthalten (*Gel-Elektrophorese*).
4. Für jeden der n Knoten v filtere alle Pfade aus, die v nicht enthalten.
5. *Ja*, falls ein Pfad verbleibt, sonst *Nein*.

Adlemans Experiment - Diskussion

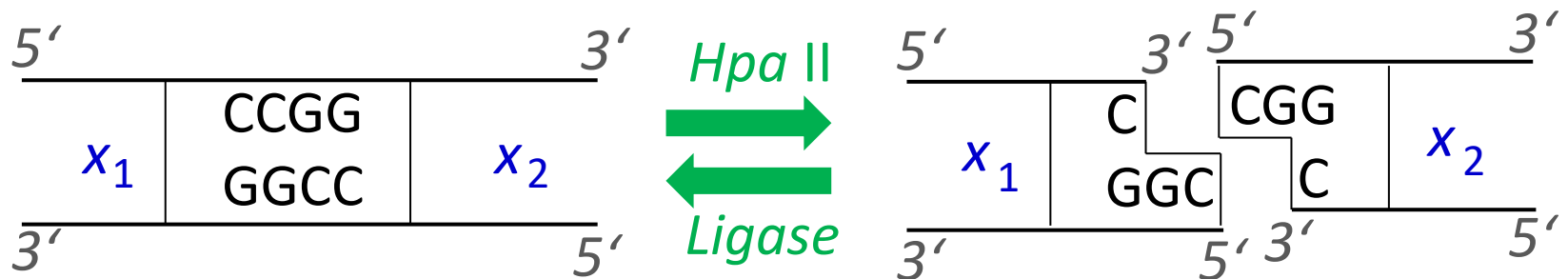
- Laufzeit (in Anzahl der Operationen): $O(n)$
 - Durchmustern erfolgt massiv parallel
 - kein Beweis für $P = NP$!
- reale Laborzeit: ca. 7 Tage
 - z.B. Vorbereitung der Filter
 - verbesserbar, z.B. durch teilweise Automatisierung
- Anzahl der benötigten Moleküle wächst rasant
 - Anzahl abhängig von Fehlerrate der Operationen
 - Undurchführbarkeit schon bei realistischen Problemgrößen
- zeigt prinzipielle Machbarkeit
 - Inspiration von vielen neuen Berechnungsmodellen

Weitere Erfolge (und Probleme)

- DNA-Algorithmus für das SAT-Problem [L]
- DNA-Algorithmus, der den DES-Code knackt [ARRW]
 - Ermittlung gültiger Schlüssel durch massiv parallele Suche
 - Faktor **Effizienz**: ca. 5 Tage bei 1 s pro Operation,
realistischer: ca. 18 Jahre bei 1 Tag pro Operation
 - Faktor **Fehlerrate**: ca. 1,4 g DNA bei Fehlerrate 10^{-4} ,
ca. 23 Erdmassen DNA bei 10^{-2}
- Ausnutzung weiterer Operationen/Enzyme
 - „Werkzeugkasten“
 - *Beispiel: Hybridisation*

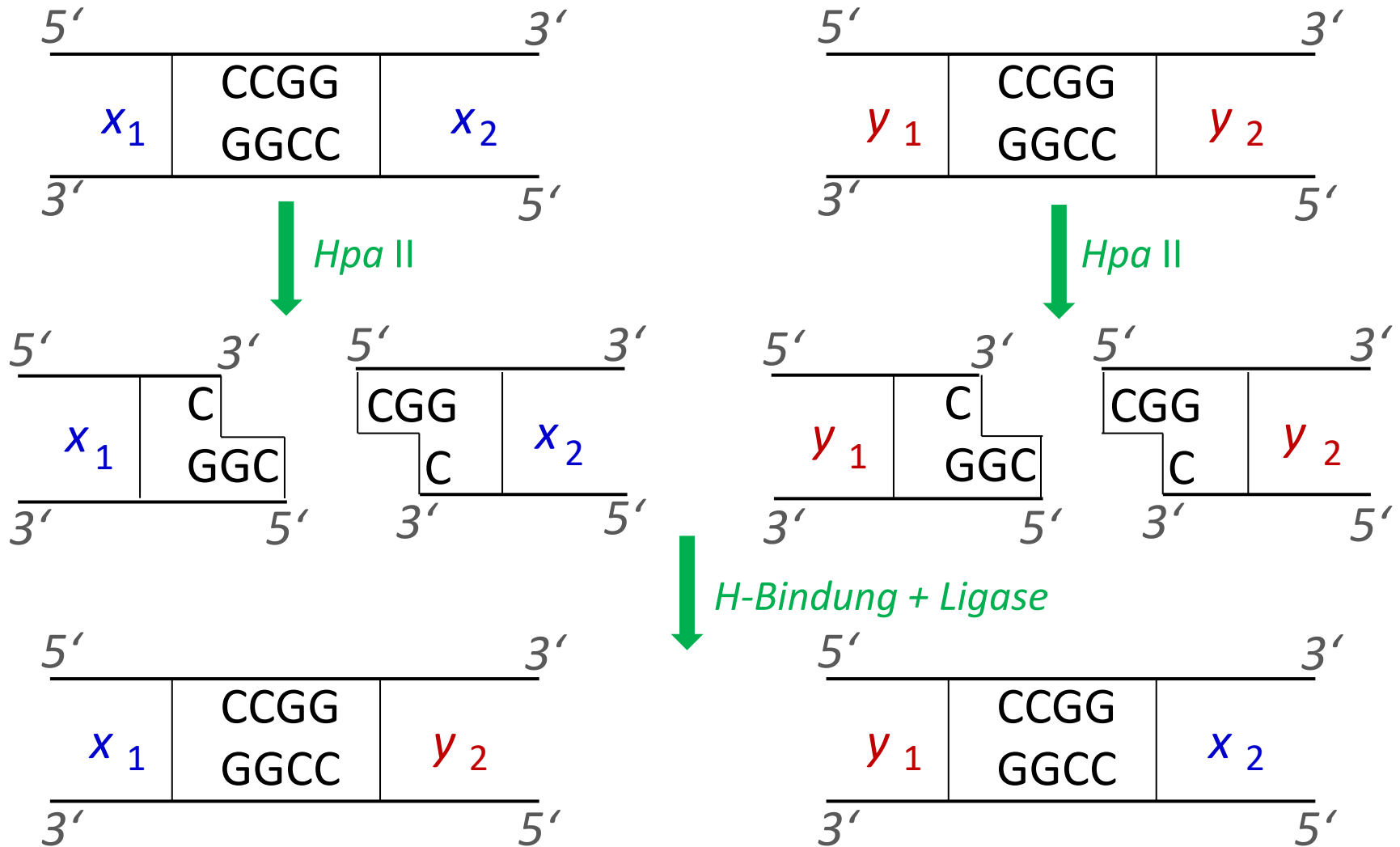
„Cut-and-Paste“ von DNA

- **Cut:** Lösen der Phosphodiesterbindung innerhalb bestimmter Nukleotid-Sequenzen
 - verschiedene Enzyme zerstören die starke chemische Bindung
 - Enzym bestimmt Position und Art des „Schnitts“



- **Paste:** Verbindung zwischen der Phosphorgruppe an 5' und der OH-Gruppe an 3' (*Ligation*)
 - Enzym: *Ligase*

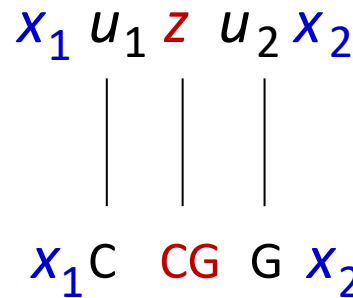
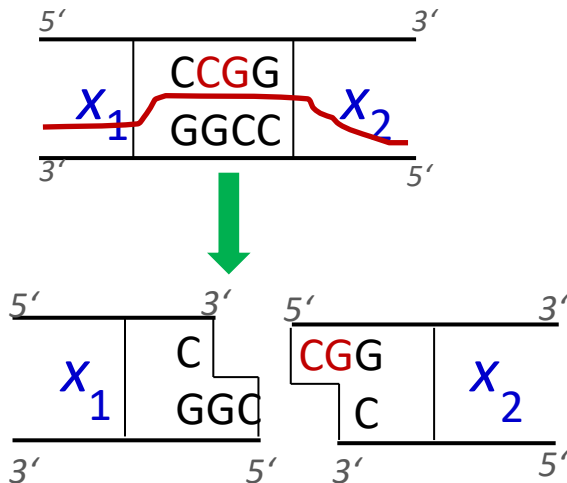
Hybridisation (Beispiel)



3.4 Abstraktion – von *in vitro* zu *in ratione*

Splicing-Systeme – Idee (1)

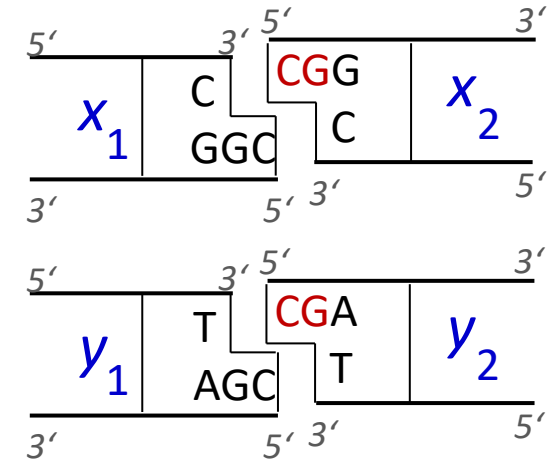
- Tom Head (1987)
- verallgemeinert enzymatisches „Cut-and-Paste“
- **Axiome** (anfängliche Strings) über einem **Alphabet**
- **Regeln:** Festlegung der Teilstrings, auf die spaltende Enzyme wirken



(C,CG,G)

Splicing-Systeme – Idee (2)

- Strings nach „Cut“ gemäß (u_1, CG, u_2)
rekombinieren mit
 Strings nach „Cut“ gemäß (v_1, CG, v_2) ,
immer wenn CG am 5'-Ende frei ist



- Regeln sind **Paare** $((u_1, z, u_2), (v_1, z, v_2))$
 oder $((u, u'), (v, v'))$

mit $u = u_1$, $u' = zu_2$,
 $v = v_1$, $v' = zv_2$

Splicing-Systeme - formal

- $G = (V, A, R)$
 - V Alphabet, $\#, \$ \notin V$
 - $A \subseteq V^*$ (Menge der Axiome)
 - R Menge von Regeln der Form $r = u_1\#u_2\$u_3\#u_4$ ($u_i \in V^*$)

- $(x, y) \xRightarrow[r]{} z$ gdw. $x = x_1 u_1 u_2 x_2$,

x_1	u_1	u_2	x_2
-------	-------	-------	-------

 $y = y_1 u_3 u_4 y_2$,

y_1	u_3	u_4	y_2
-------	-------	-------	-------

 $z = x_1 u_1 u_4 y_2$.

x_1	u_1	u_4	y_2
-------	-------	-------	-------

Sprachen von Splicing-Systemen

- $G = (V, A, R)$
- **nicht iteriertes Splicing:**

$$L(G) = \{ z \in V^* \mid (x, y) \xRightarrow[r]{} z, \quad x, y \in A, \quad r \in R \}$$

$$\mathcal{L}_1(\mathfrak{F}, \mathcal{R}) = \{ L(G) \mid G = (V, A, R) \text{ mit } A \in \mathfrak{F}, \quad R \in \mathcal{R} \}$$

($\mathfrak{F}, \mathcal{R}$ Sprachfamilien)

Splicing-Systeme - Einordnung

$$\mathcal{L}_1(\mathcal{F}, \mathcal{R}) :$$

$\mathcal{F} \backslash \mathcal{R}$	FIN	REG	LIN	CF	CS	RE
FIN	FIN	FIN	FIN	FIN	FIN	FIN
REG	REG	REG	REG,LIN	REG,CF	REG,RE	REG,RE
LIN	LIN,CF	LIN,CF	RE	RE	RE	RE
CF	CF	CF	RE	RE	RE	RE
CS	RE	RE	RE	RE	RE	RE
RE	RE	RE	RE	RE	RE	RE

Splicing-Systeme - Erweiterungen

- **Iteration**

- bei Vorhandensein gewisser Enzyme (u.a. Ligase):
Splicing Prozess stoppt nicht
- Rekombinate als neue Ausgangsprodukte

- **Hilfssymbole**

- realisierbar durch Filtern
- theoretisch interessante Resultate

Iteriertes Splicing

$$\mathcal{H}(\mathcal{F}, \mathcal{R}) :$$

$\mathcal{F} \backslash \mathcal{R}$	FIN	REG	LIN	CF	CS	RE
FIN	FIN,REG	FIN,RE	FIN,RE	FIN,RE	FIN,RE	FIN,RE
REG	REG	REG,RE	REG,RE	REG,RE	REG,RE	REG,RE
LIN	LIN,CF	LIN,RE	LIN,RE	LIN,RE	LIN,RE	LIN,RE
CF	CF	CF,RE	CF,RE	CF,RE	CF,RE	CF,RE
CS	CS,RE	CS,RE	CS,RE	CS,RE	CS,RE	CS,RE
RE	RE	RE	RE	RE	RE	RE

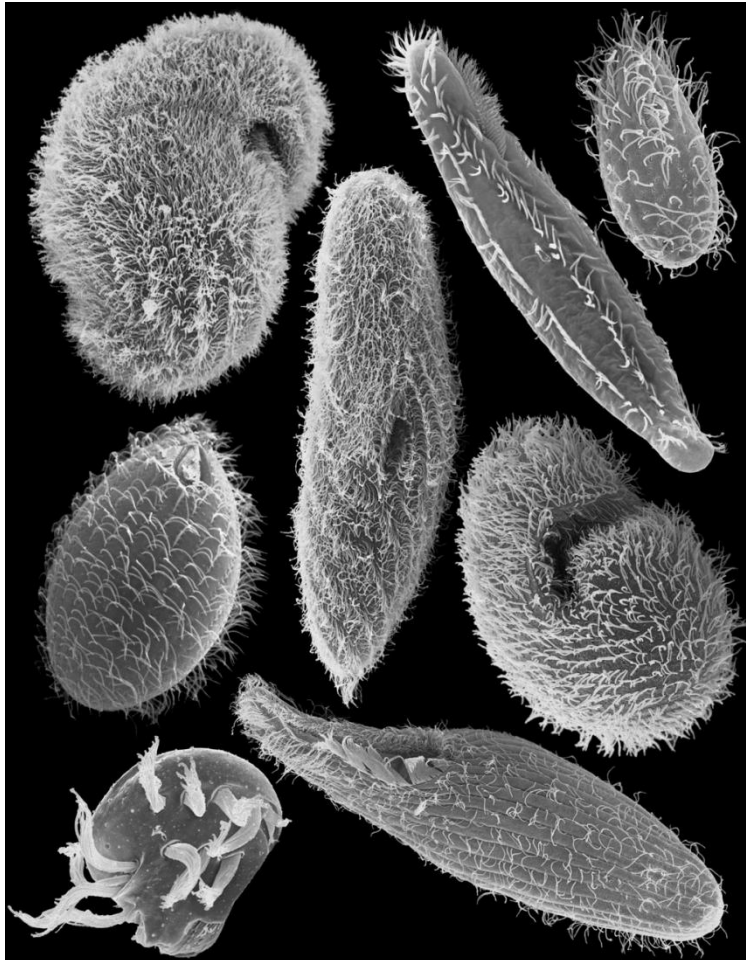
Iteriertes Splicing mit Hilfssymbolen

$$\mathcal{EH}(\mathcal{I}, \mathcal{R}) :$$

$\mathcal{I} \backslash \mathcal{R}$	FIN	REG	LIN	CF	CS	RE
FIN	REG	RE	RE	RE	RE	RE
REG	REG	RE	RE	RE	RE	RE
LIN	LIN,CF	RE	RE	RE	RE	RE
CF	CF	RE	RE	RE	RE	RE
CS	RE	RE	RE	RE	RE	RE
RE	RE	RE	RE	RE	RE	RE

3.5 Noch einmal *in vivo* - Können lebende Zellen rechnen?

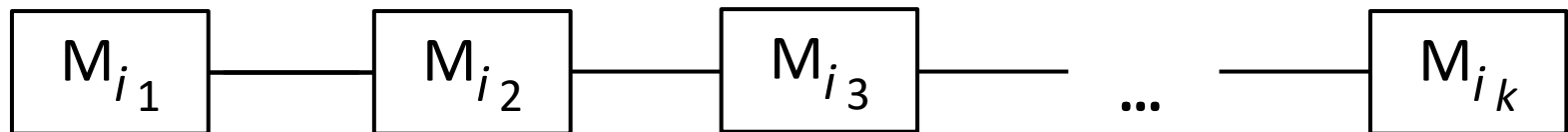
Wimperntierchen (Ciliate)



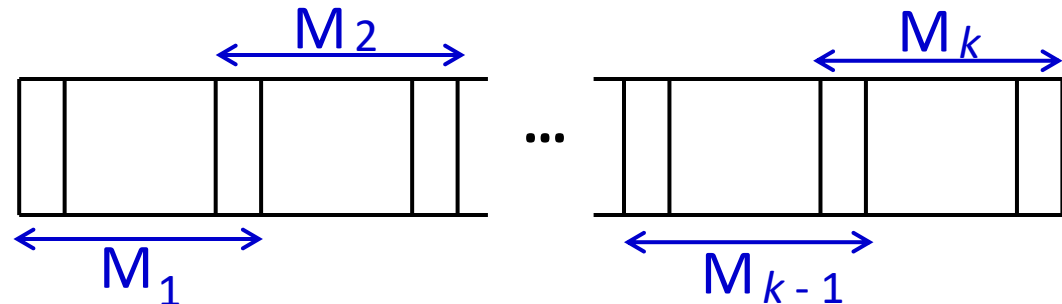
- Einzeller
- zwei verschiedene Zellkerne:
 1. **Mikronukleus**
 - steuert die Fortpflanzung
 2. **Makronukleus**
 - steuert das Soma der Zelle
 - löst sich während der Zellteilung auf
 - **wird aus Mikronukleus neu gebildet**

DNA-Struktur bei Ciliaten

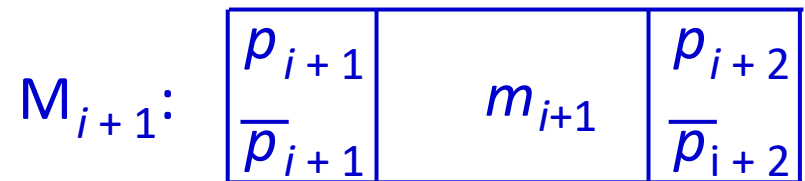
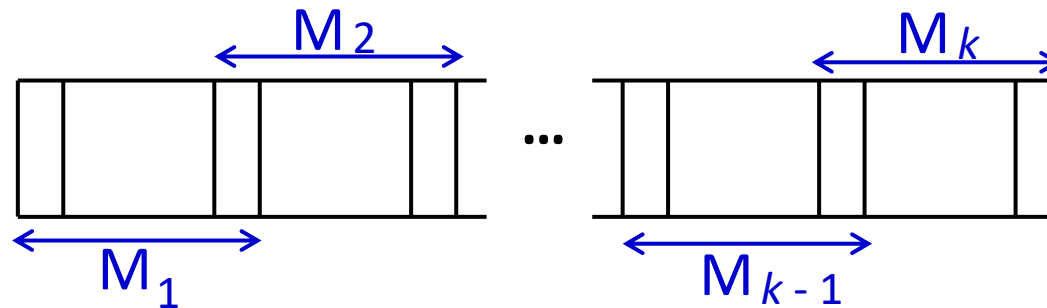
- im Mikronukleus:
 - **MDS**-Stücke (*macronuclear destined sequences*):
speichern die eigentliche genetische Information
 - **IES** (*internally eliminated sequences*):
ohne genetische Information; separieren MDS-Teile



- im Makronukleus:
 - nur MDS, „sortiert“



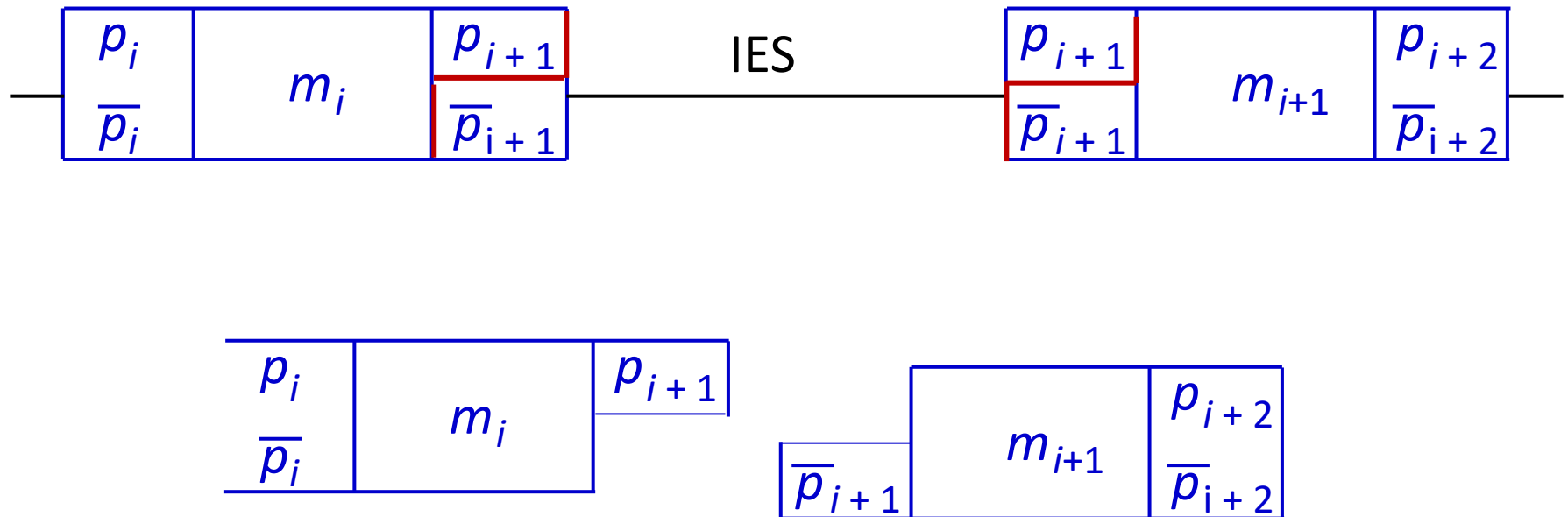
MDS-Struktur



Pointer

übereinstimmende
Pointer

MDS im Mikronukleus



Funktioniert bei *direkter* Wiederholung der Pointer.

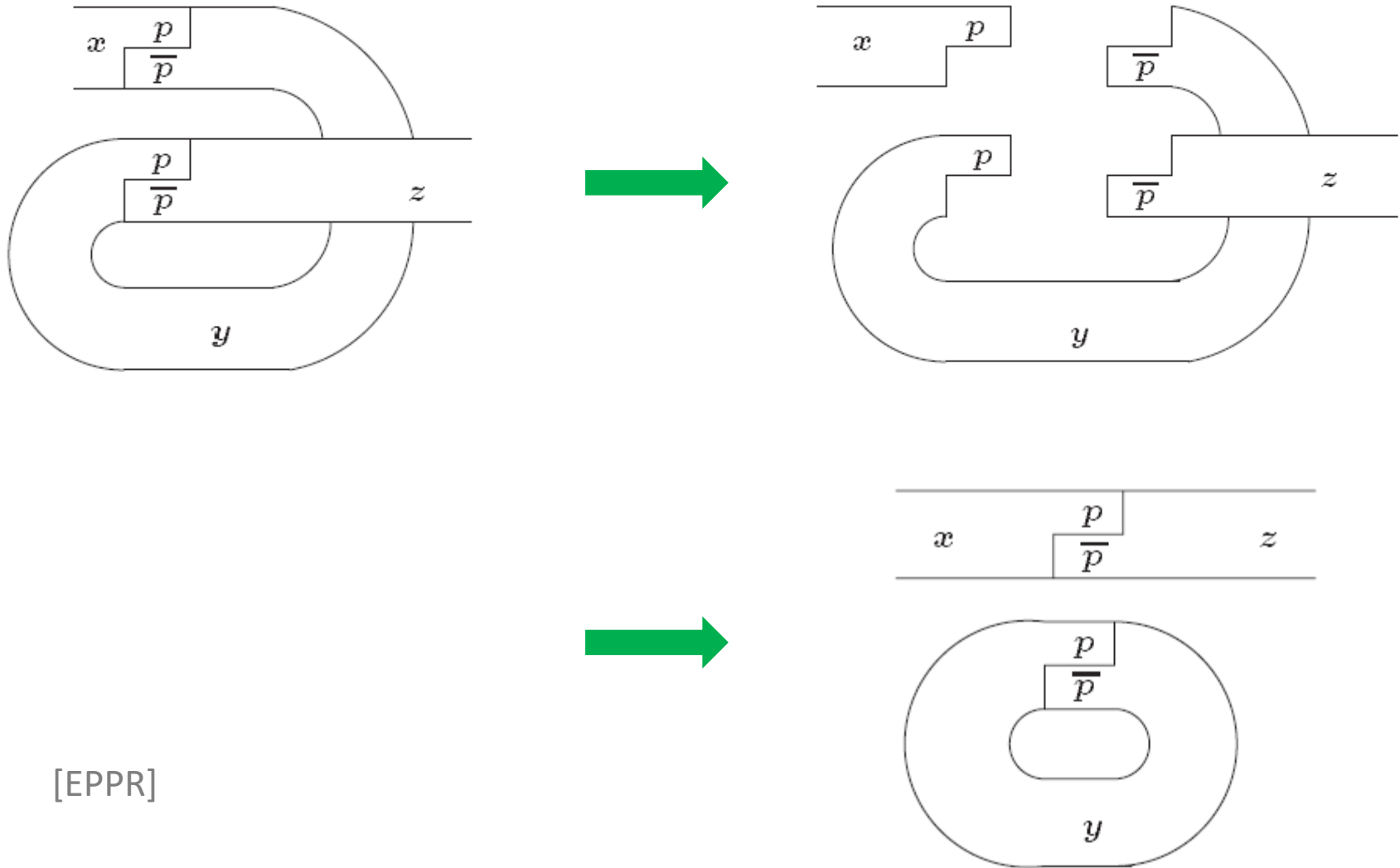
Umordnungen und **Spiegelungen** sind erforderlich.

Die Assembly-Operationen

1. **ld** (loop, direct repeat)-excision
2. **hi** (hairpin, inverted repeat)-excision/reinsertion
3. **dlad** (double loop, alternating direct repeat)-excision/reinsertion

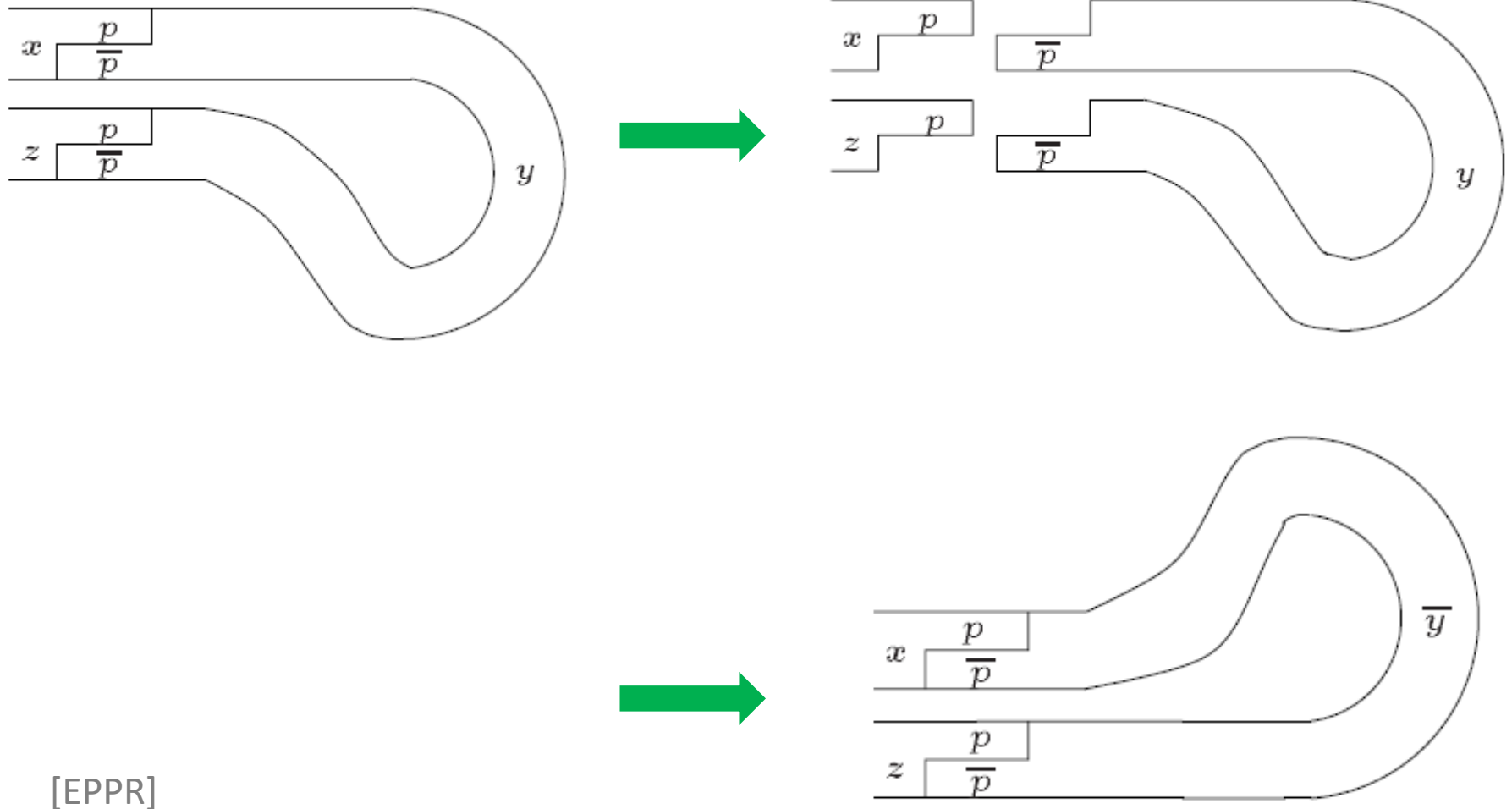
Genügen, um den Makronukleus aus dem Mikronukleus zu generieren.

Id - Operation



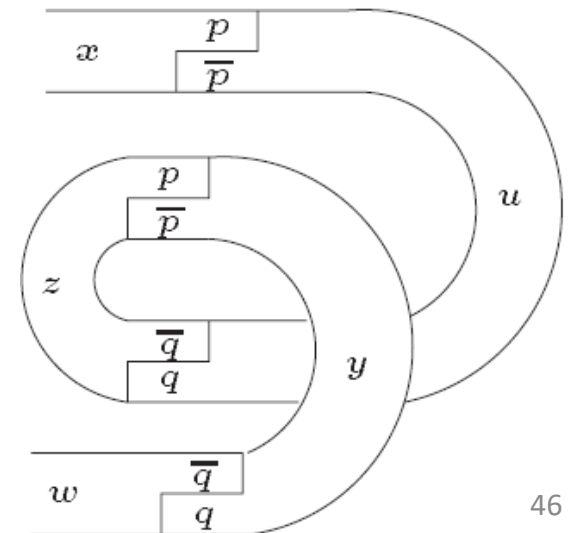
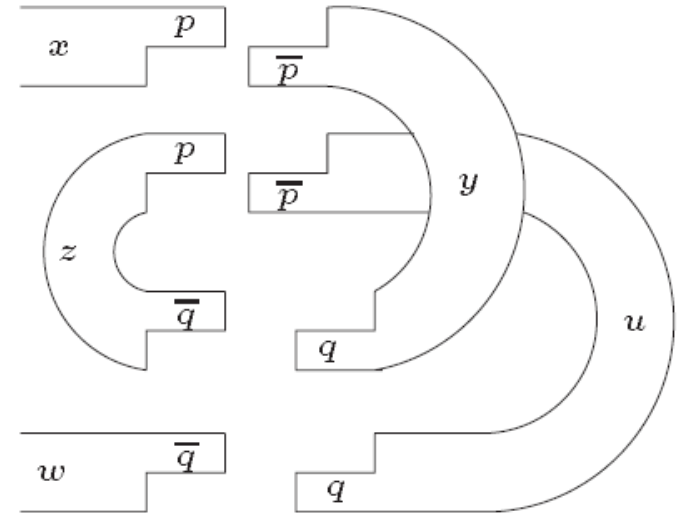
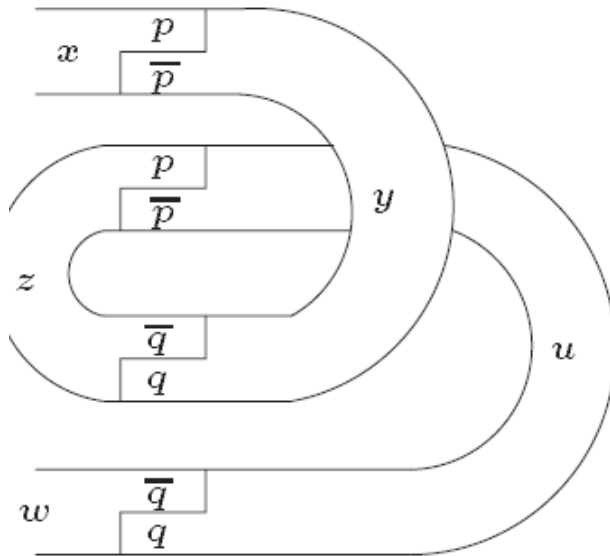
[EPPR]

hi - Operation



[EPPR]

dlad - Operation



[EPPR]

Die Herausforderung

- Entdecken eines neuen, weiteren Verständnisses von Berechenbarkeit / Computation
- Wahrnehmen der uns umgebenden Welt im Sinne der Informationsverarbeitung
- „*Biology and Computer Science – life and computation are related. I am confident that at their interface great discoveries await those who seek them.*“
(Adelman 1998)

Referenzen

- [A] L. M. Adleman: Molecular computation of solutions to combinatorial problems. *Science*, 226 (1994), 1021–1024.
- [ARRW] L. M. Adleman, P. W. K. Rothemund, S. Roweiss, E. Winfree: On applying molecular computation to the Data Encrypted Standard. In E. Baum et al. (Hrsg.): *DNA Based Computers*. Proc. Second Annual Meeting, Princeton, 1996.
- [EPPR] A. Ehrenfeucht, I. Petre, D.M. Prescott, G. Rozenberg: *Universal and Simple Operations for Gene Assembly in Ciliates. Where Mathematics, Computer Science, Linguistics and Biology Meet*. Kluwer Academic Publishers, Norwell, MA 2001.
- [L] R. J. Lipton: Using DNA to solve NP-complete problems. *Science*, 268 (1995), 542-545.
- [PRS] Gh. Păun, G. Rozenberg, A. Salomaa: *DNA Computing. New Computing Paradigms*. Springer-Verlag, Berlin – Heidelberg, 1998.

Vielen DNAk !