



openHPI
Einführung in die Mathematik der Algorithmik
– Dijkstras Algorithmus

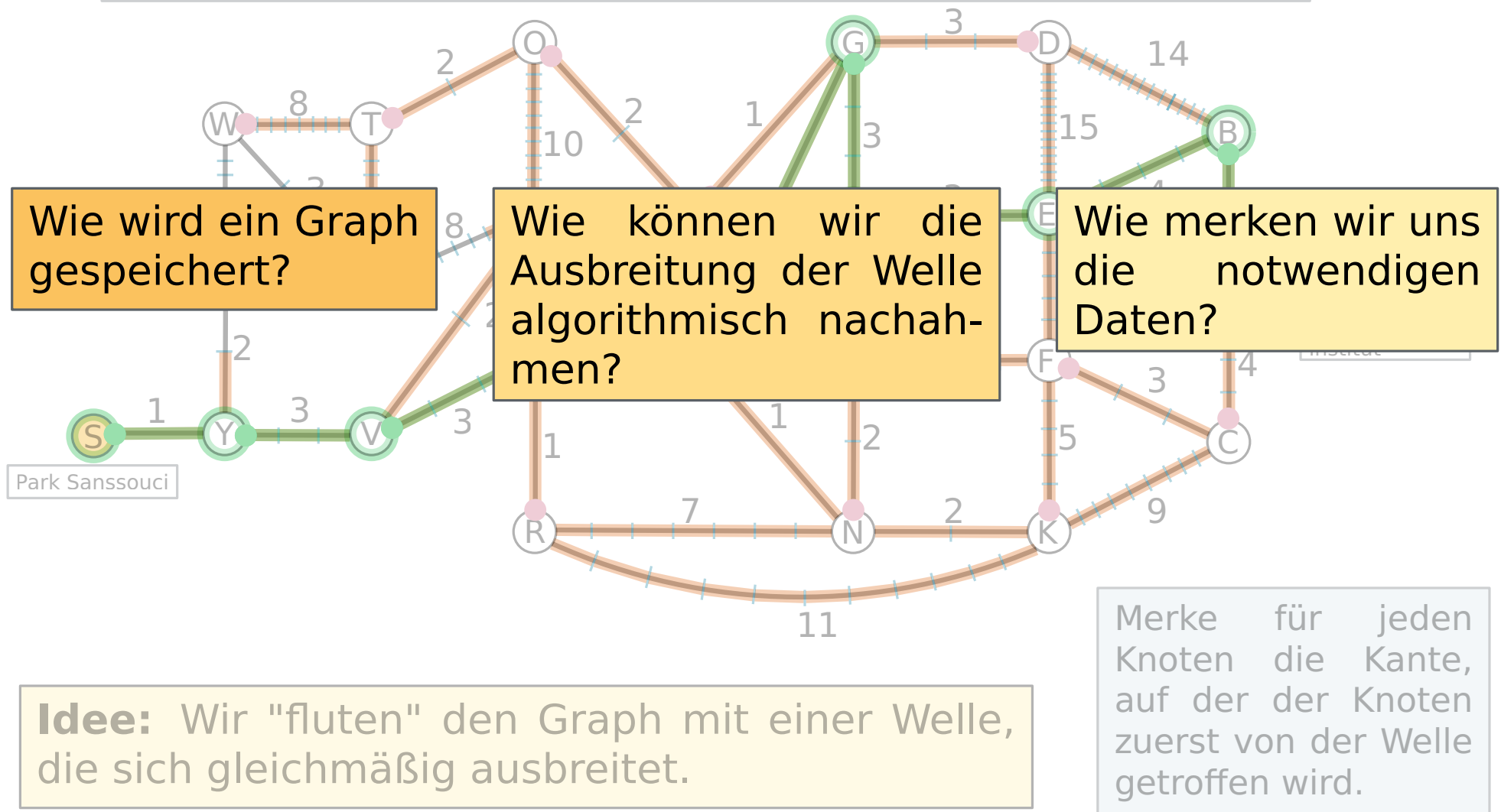
Timo Kötzing und Pascal Lenzner

Hasso-Plattner-Institut

Universität Potsdam

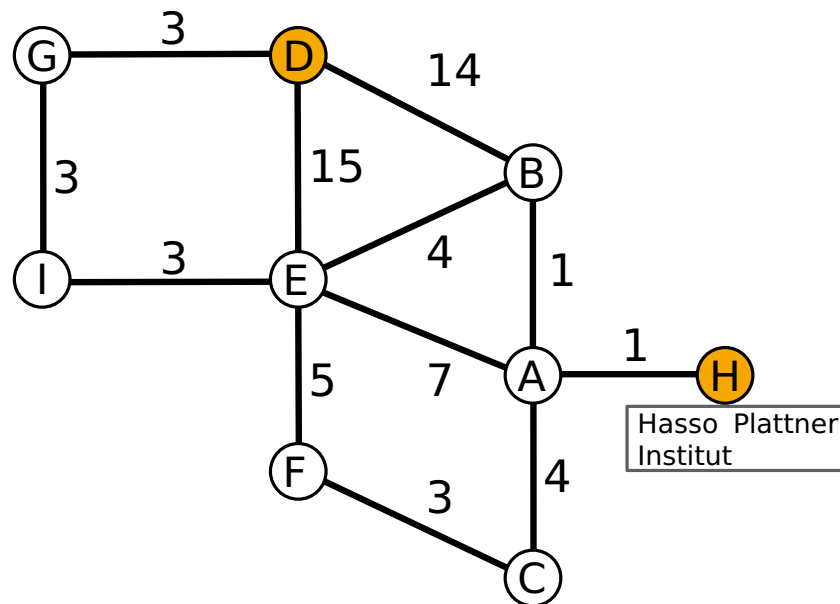
Den kürzesten Pfad in einem Graph finden...

Rekonstruiere kürzesten Pfad mit Hilfe der gemerkten Information.



Wie wird ein Graph gespeichert?

Adjazenzliste



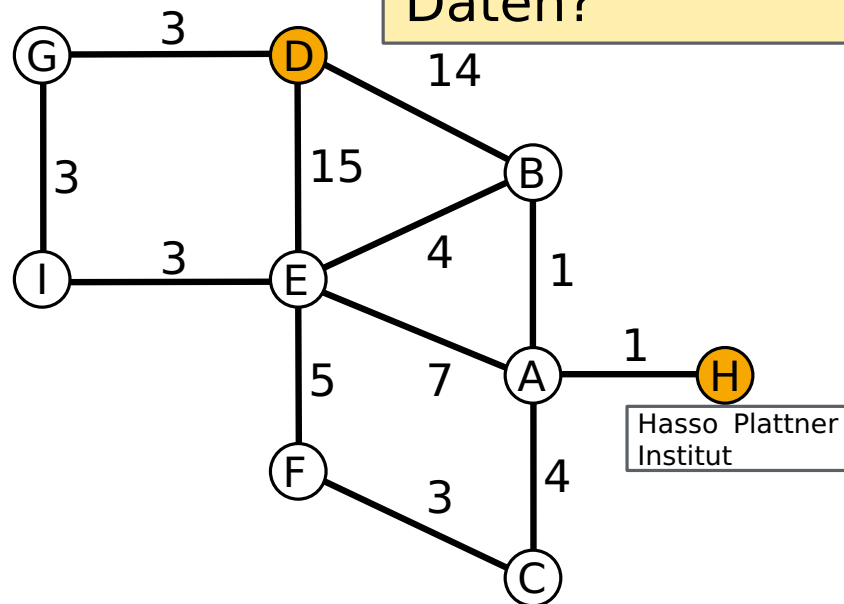
- benötigen für jeden Knoten Informationen über die Kanten zu den Nachbarknoten
- merken uns pro Knoten die Nachbarn und die Länge der jeweiligen Kanten

Knoten	Nachbarn mit Länge der Kante
(A)	(B,1),(C,4),(E,7),(H,1)
(B)	(A,1),(E,4),(D,14)
(C)	(A,4),(F,3)
(D)	(B,14),(E,15),(G,3)
(E)	(A,7),(B,4),(D,15),(F,5),(I,3)
(F)	(C,3),(E,5)
(G)	(D,3),(I,3)
(H)	(A,1)
(I)	(E,3),(G,3)

Wie können wir die Ausbreitung der Welle algorithmisch nachahmen?

Idee: Überlege welcher Knoten als nächster von der Welle getroffen wird.

Wie merken wir uns die notwendigen Daten?



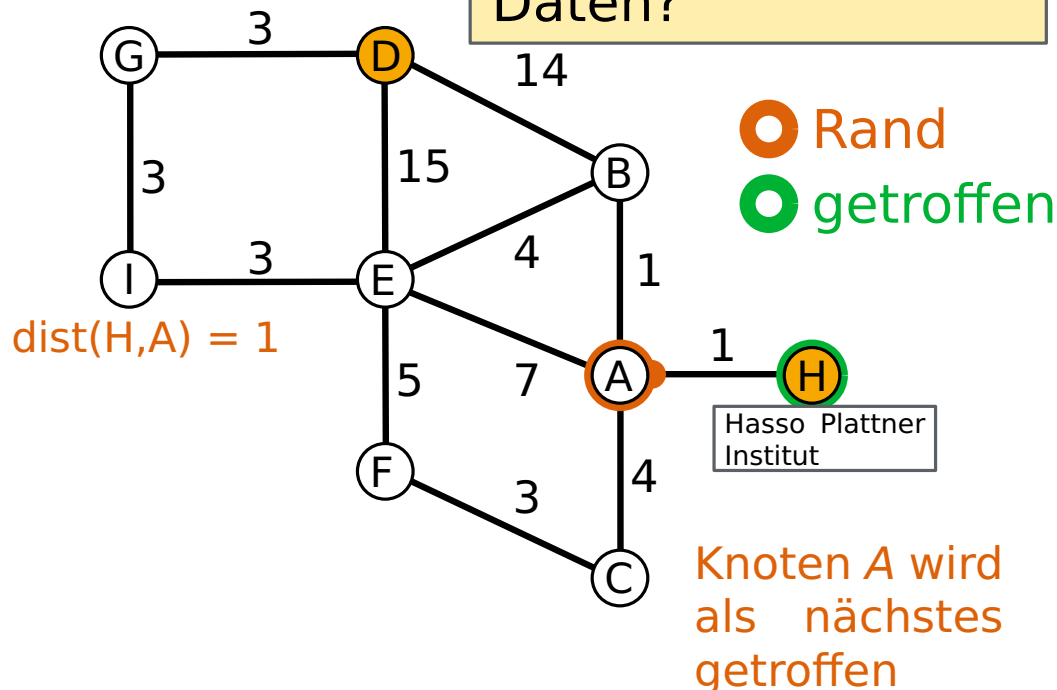
- merke welcher Knoten schon getroffen wurde (merke auch durch welche Kante)
- betrachte **Randknoten**: Nachbarn von getroffenen Knoten, die noch nicht getroffen wurden
- aktualisiere Distanz von Randknoten zum Startknoten
- als nächstes wird der Randknoten getroffen, dessen **Distanz zum Startknoten minimal** ist!
- iteriere bis alle Knoten getroffen wurden

Den kürzesten Pfad in einem Graph finden...

Wie können wir die Ausbreitung der Welle algorithmisch nachahmen?

Idee: Überlege welcher Knoten als nächster von der Welle getroffen wird.

Wie merken wir uns die notwendigen Daten?



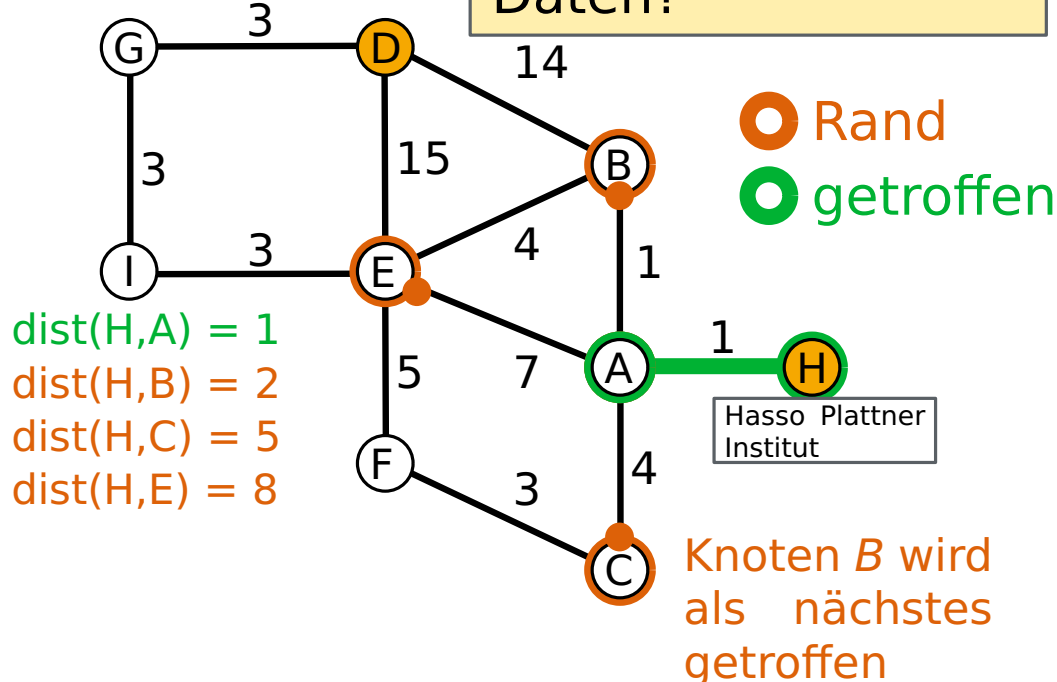
- merke welcher Knoten schon getroffen wurde (merke auch durch welche Kante)
- betrachte **Randknoten**: Nachbarn von getroffenen Knoten, die noch nicht getroffen wurden
- aktualisiere Distanz von Randknoten zum Startknoten
- als nächstes wird der Randknoten getroffen, dessen **Distanz zum Startknoten minimal** ist!
- iteriere bis alle Knoten getroffen wurden

Den kürzesten Pfad in einem Graph finden...

Wie können wir die Ausbreitung der Welle algorithmisch nachahmen?

Idee: Überlege welcher Knoten als nächster von der Welle getroffen wird.

Wie merken wir uns die notwendigen Daten?



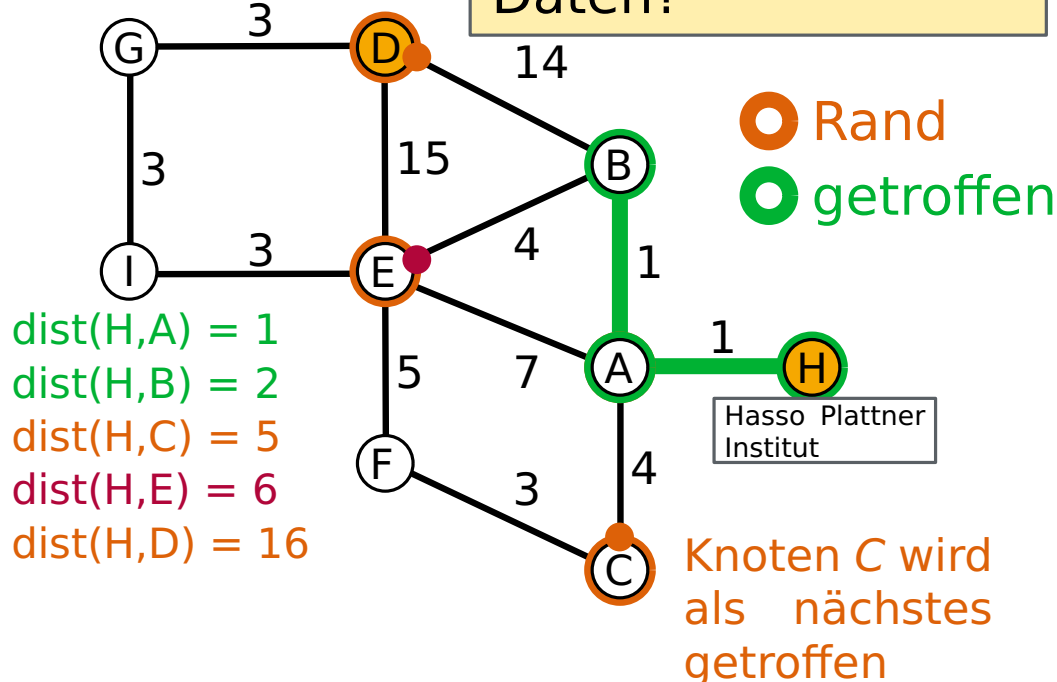
- merke welcher Knoten schon getroffen wurde (merke auch durch welche Kante)
- betrachte **Randknoten**: Nachbarn von getroffenen Knoten, die noch nicht getroffen wurden
- aktualisiere Distanz von Randknoten zum Startknoten
- als nächstes wird der Randknoten getroffen, dessen **Distanz zum Startknoten minimal** ist!
- iteriere bis alle Knoten getroffen wurden

Den kürzesten Pfad in einem Graph finden...

Wie können wir die Ausbreitung der Welle algorithmisch nachahmen?

Idee: Überlege welcher Knoten als nächster von der Welle getroffen wird.

Wie merken wir uns die notwendigen Daten?



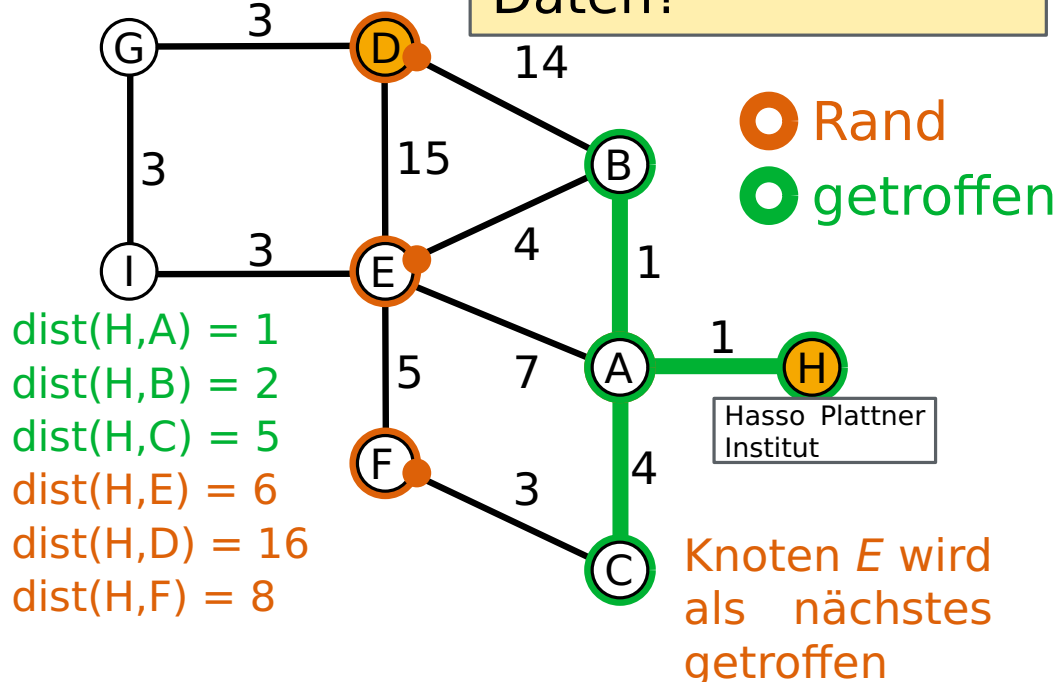
- merke welcher Knoten schon getroffen wurde (merke auch durch welche Kante)
- betrachte **Randknoten**: Nachbarn von getroffenen Knoten, die noch nicht getroffen wurden
- aktualisiere Distanz von Randknoten zum Startknoten
- als nächstes wird der Randknoten getroffen, dessen **Distanz zum Startknoten minimal** ist!
- iteriere bis alle Knoten getroffen wurden

Den kürzesten Pfad in einem Graph finden...

Wie können wir die Ausbreitung der Welle algorithmisch nachahmen?

Idee: Überlege welcher Knoten als nächster von der Welle getroffen wird.

Wie merken wir uns die notwendigen Daten?



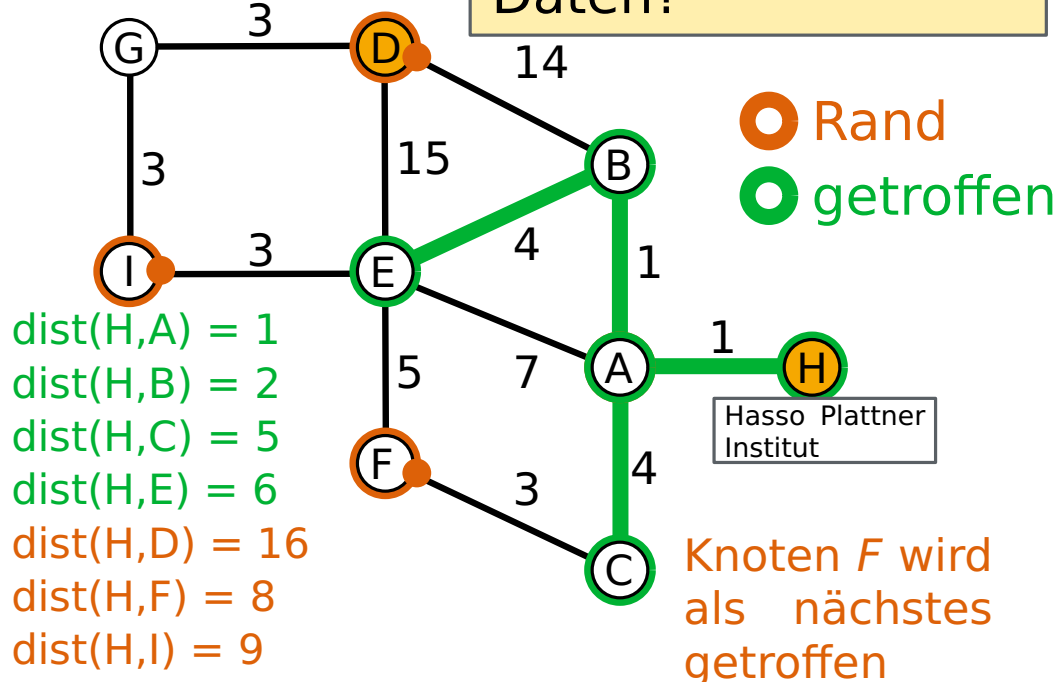
- merke welcher Knoten schon getroffen wurde (merke auch durch welche Kante)
- betrachte **Randknoten**: Nachbarn von getroffenen Knoten, die noch nicht getroffen wurden
- aktualisiere Distanz von Randknoten zum Startknoten
- als nächstes wird der Randknoten getroffen, dessen **Distanz zum Startknoten minimal** ist!
- iteriere bis alle Knoten getroffen wurden

Den kürzesten Pfad in einem Graph finden...

Wie können wir die Ausbreitung der Welle algorithmisch nachahmen?

Idee: Überlege welcher Knoten als nächster von der Welle getroffen wird.

Wie merken wir uns die notwendigen Daten?



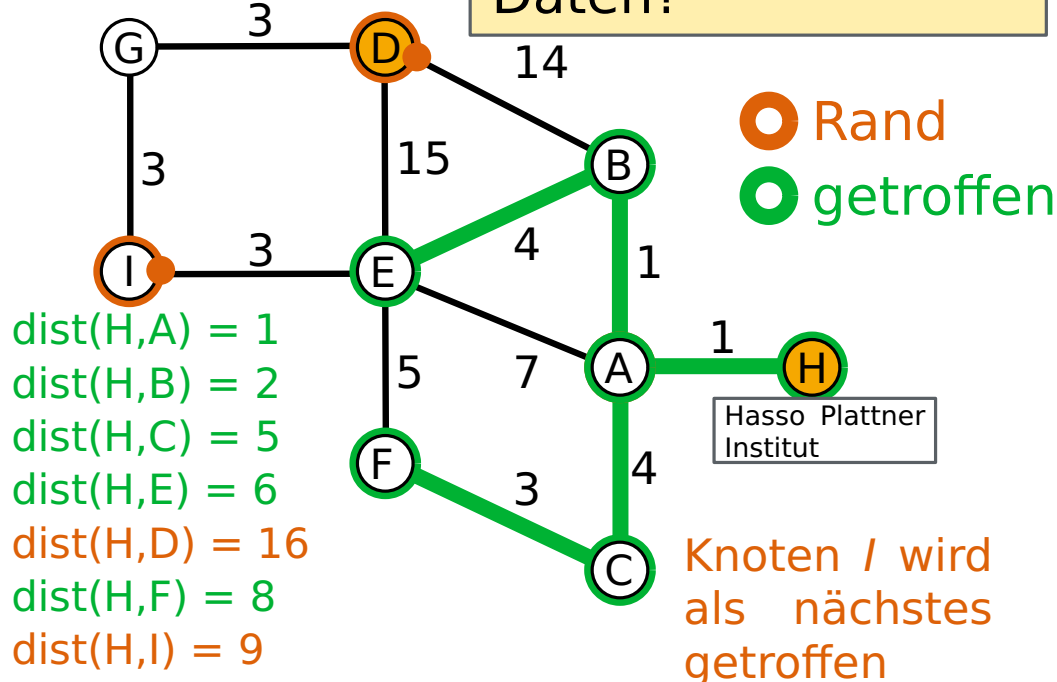
- merke welcher Knoten schon getroffen wurde (merke auch durch welche Kante)
- betrachte **Randknoten**: Nachbarn von getroffenen Knoten, die noch nicht getroffen wurden
- aktualisiere Distanz von Randknoten zum Startknoten
- als nächstes wird der Randknoten getroffen, dessen **Distanz zum Startknoten minimal** ist!
- iteriere bis alle Knoten getroffen wurden

Den kürzesten Pfad in einem Graph finden...

Wie können wir die Ausbreitung der Welle algorithmisch nachahmen?

Idee: Überlege welcher Knoten als nächster von der Welle getroffen wird.

Wie merken wir uns die notwendigen Daten?



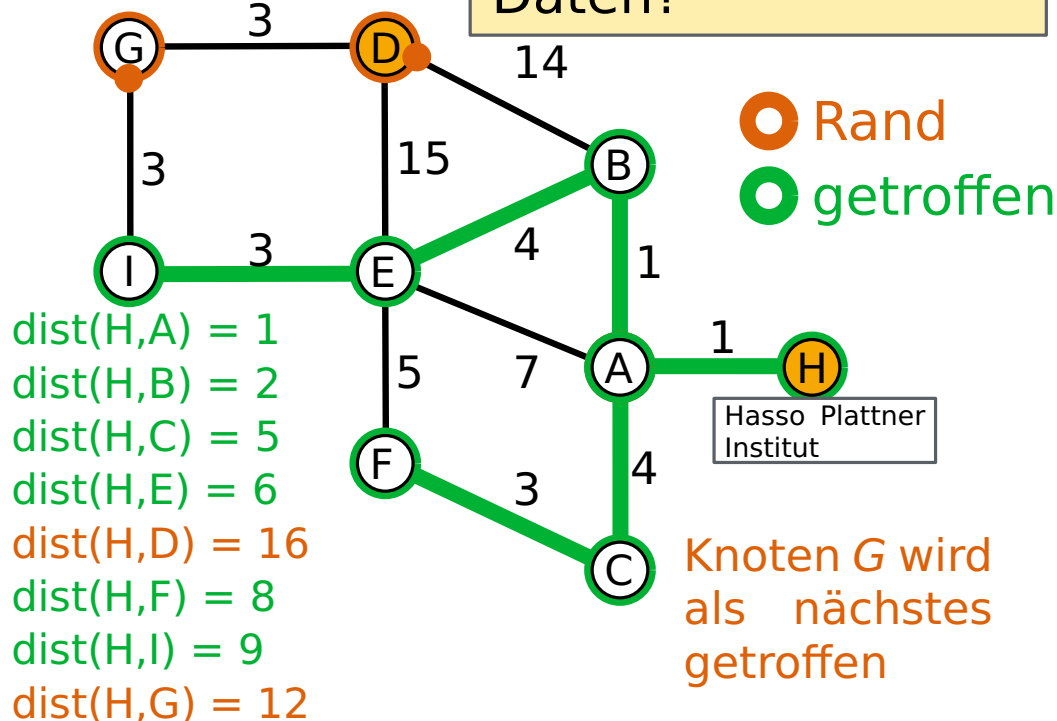
- merke welcher Knoten schon getroffen wurde (merke auch durch welche Kante)
- betrachte **Randknoten**: Nachbarn von getroffenen Knoten, die noch nicht getroffen wurden
- aktualisiere Distanz von Randknoten zum Startknoten
- als nächstes wird der Randknoten getroffen, dessen **Distanz zum Startknoten minimal** ist!
- iteriere bis alle Knoten getroffen wurden

Den kürzesten Pfad in einem Graph finden...

Wie können wir die Ausbreitung der Welle algorithmisch nachahmen?

Idee: Überlege welcher Knoten als nächster von der Welle getroffen wird.

Wie merken wir uns die notwendigen Daten?



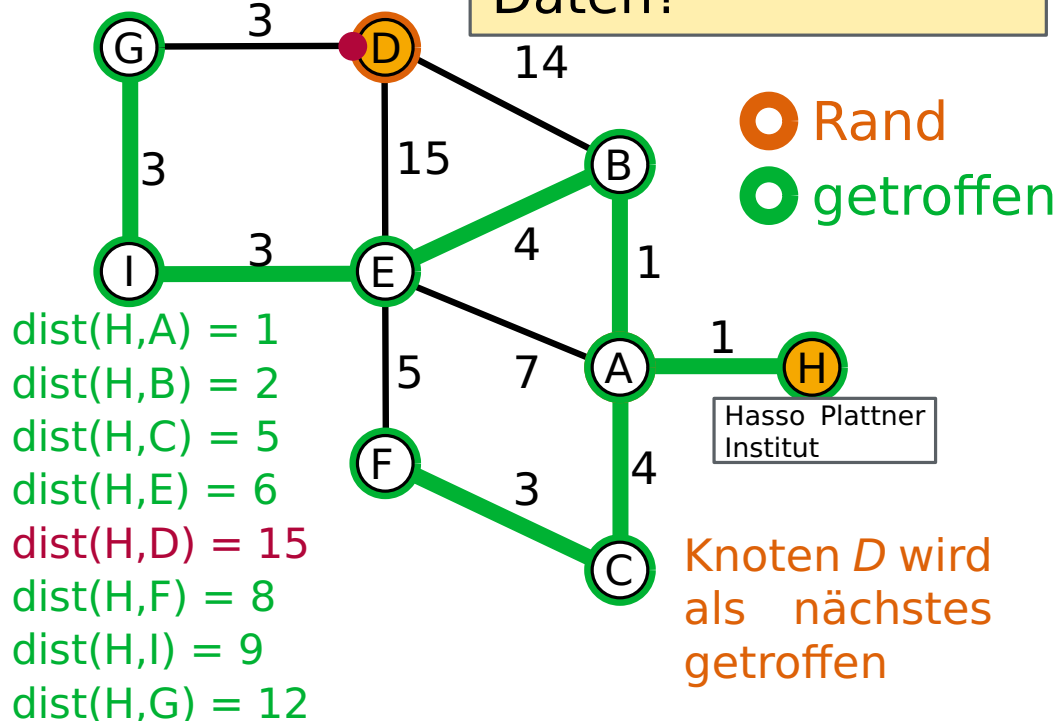
- merke welcher Knoten schon getroffen wurde (merke auch durch welche Kante)
- betrachte **Randknoten**: Nachbarn von getroffenen Knoten, die noch nicht getroffen wurden
- aktualisiere Distanz von Randknoten zum Startknoten
- als nächstes wird der Randknoten getroffen, dessen **Distanz zum Startknoten minimal** ist!
- iteriere bis alle Knoten getroffen wurden

Den kürzesten Pfad in einem Graph finden...

Wie können wir die Ausbreitung der Welle algorithmisch nachahmen?

Idee: Überlege welcher Knoten als nächster von der Welle getroffen wird.

Wie merken wir uns die notwendigen Daten?

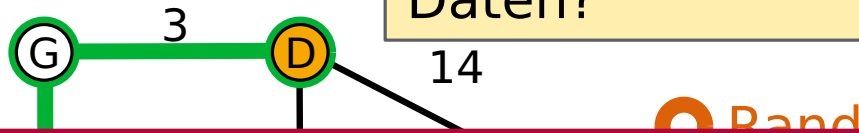


- merke welcher Knoten schon getroffen wurde (merke auch durch welche Kante)
- betrachte **Randknoten**: Nachbarn von getroffenen Knoten, die noch nicht getroffen wurden
- aktualisiere Distanz von Randknoten zum Startknoten
- als nächstes wird der Randknoten getroffen, dessen **Distanz zum Startknoten minimal** ist!
- iteriere bis alle Knoten getroffen wurden

Wie können wir die Ausbreitung der Welle algorithmisch nachahmen?

Idee: Überlege welcher Knoten als nächster von der Welle getroffen wird.

Wie merken wir uns die notwendigen Daten?



Kürzester Pfade Algorithmus von Dijkstra

- merke welcher Knoten schon getroffen wurde (merke auch durch welche Kante)
- betrachte **Randknoten**: Nachbarn von getroffenen Knoten, die noch nicht getroffen wurden
- aktualisiere Distanz von Randknoten zum Startknoten
- als nächstes wird der Randknoten getroffen, dessen **Distanz zum Startknoten minimal** ist!
- iteriere bis alle Knoten getroffen wurden

$\text{dist}(H,I) = 9$
 $\text{dist}(H,G) = 12$

fertig!