Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Sprachverarbeitung I / 12 HS 2016

Spracherkennung mittels Mustervergleich

Buch: Kapitel 12

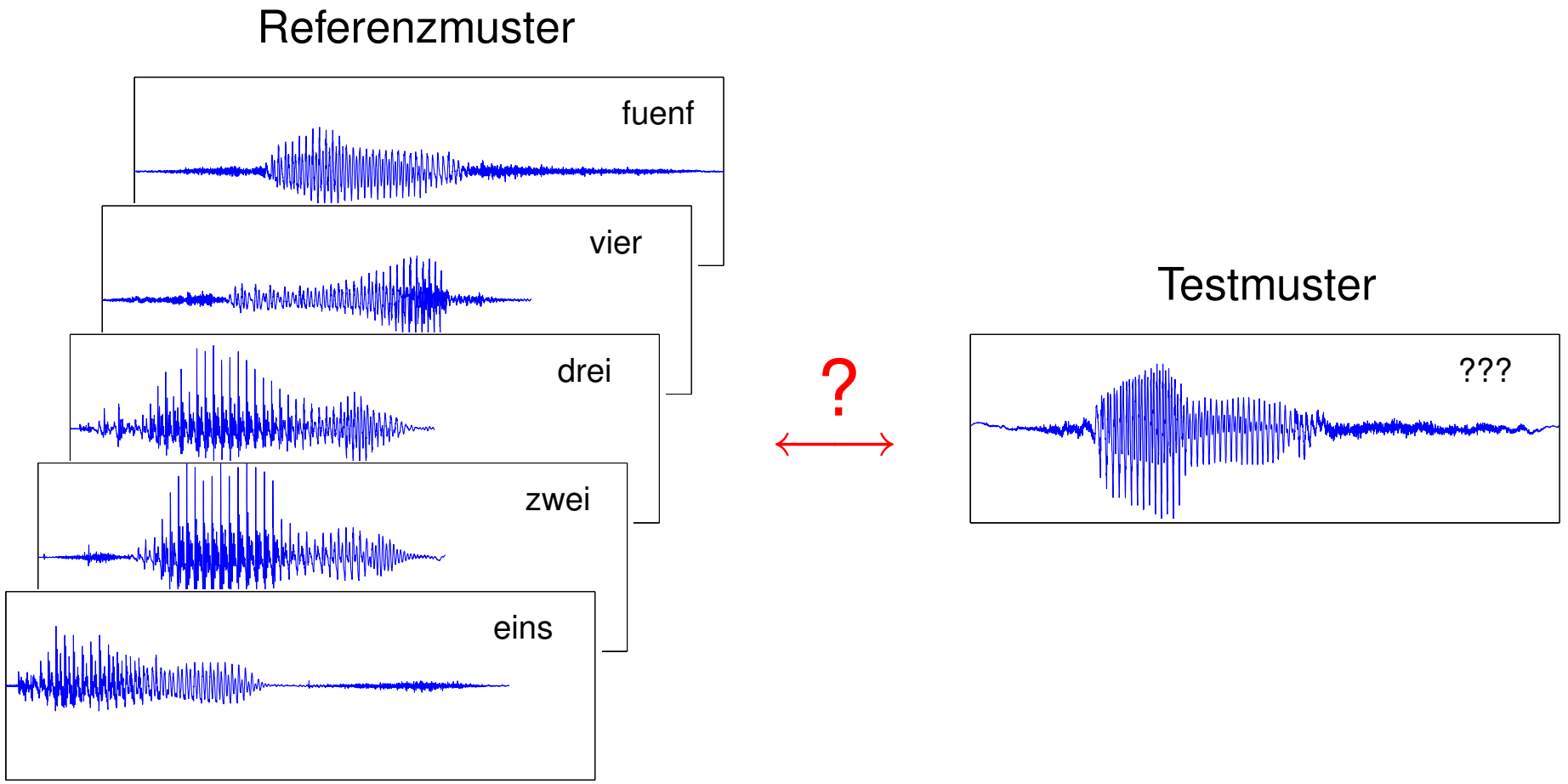
Beat Pfister

Programm heute:

Vorlesung:
- Prinzip des Sprachmustervergleichs
- Zeitanpassung von Sprachmustern

Übung:
- ⋆ Dynamische Zeitanpassung

Spracherkennung mittels Mustervergleich



Spracherkennung mittels Mustervergleich

Tatsache: Wenn eine Person ein Wort zweimal spricht, dann sind die beiden Sprachsignale $s(w_1)$ und $s(w_2)$ nicht gleich.

Problem: Mustervergleich kann nicht auf Gleichheit testen!

Ansatz: Ermitteln des Unterschiedes: Distanz $D\{s(w_1), s(w_2)\}$

mit $D\{s(w_1), s(w_2)\} \rightarrow$ klein, falls $w_1 = w_2$

und $D\{s(w_1), s(w_2)\} \rightarrow$ gross, falls $w_1 \neq w_2$

Wie ist die Distanz zweckmässig zu ermitteln?

Beobachtung: Wenn eine Person ein Wort zweimal spricht, dann unterscheiden sich die Äusserungen hauptsächlich in der Prosodie: Lautheit, Grundfrequenz, Dauer / Sprechrhythmus

Anforderung: Distanz $D(s_x, s_y)$ soll durch prosodische Unterschiede möglichst wenig beeinfusst werden

Vorgehen:

1. Verwendung geeigneter Sprachmerkmale: MFCC >>>
   - $\bar{c}(0)$ kann weggelassen werden
   - Einfluss von F_0 ist gering
2. Distanzmass >>>

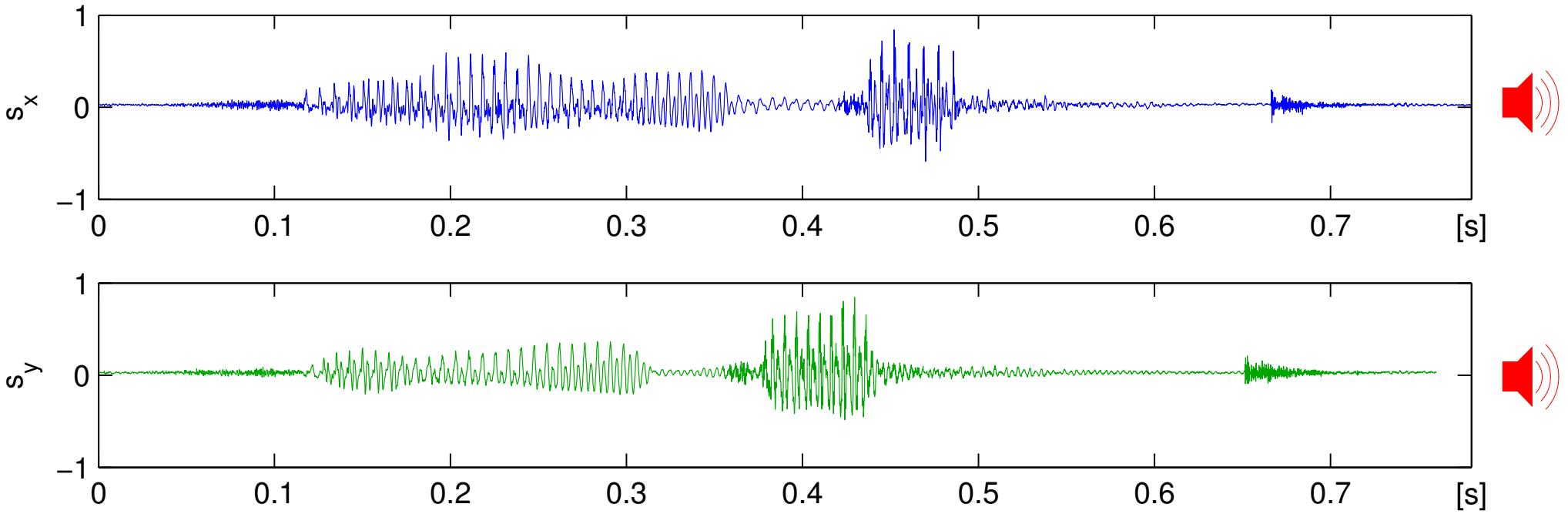
Vergleich von Sprachmustern

Ermitteln der Gesamtdistanz $D(\mathbf{X}, \mathbf{Y})$:

Addieren der lokalen Distanzen $d(i, j)$

Problem: $\mathbf{X}$ und $\mathbf{Y}$ sind meistens nicht gleich lang

Vergleich von Sprachmustern



Problem: zeitliche Struktur verschieden!

Lösung: nichtlineare zeitliche Anpassung

Zeitliche Anpassung von Sprachmustern

Gegeben: Zwei Sequenzen von Merkmalsvektoren (Muster)
$\mathbf{X} = \mathbf{x}_1\mathbf{x}_2 \ldots \mathbf{x}_{T_X}$ und $\mathbf{Y} = \mathbf{y}_1\mathbf{y}_2 \ldots \mathbf{y}_{T_Y}$

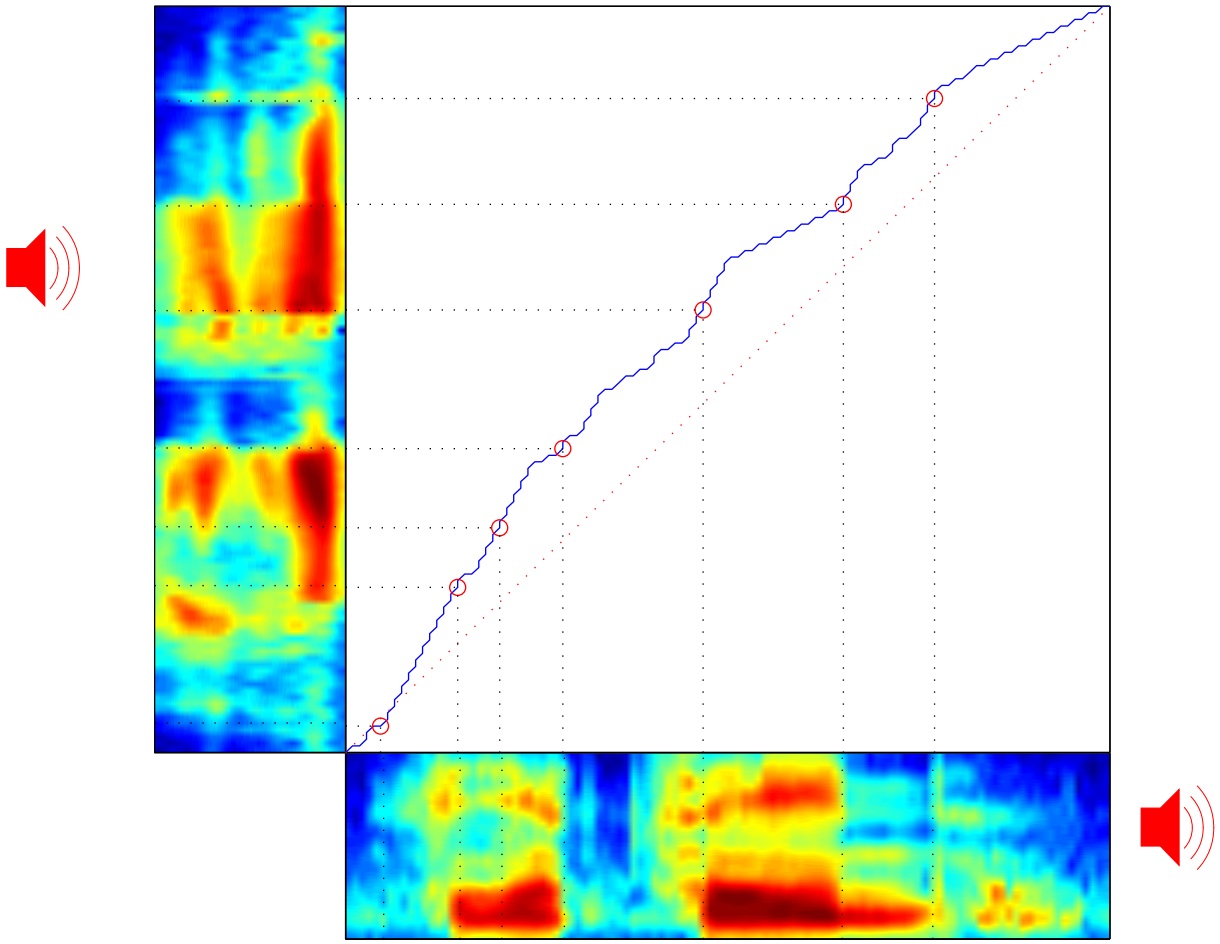
Gesucht: Optimale Zeitanpassung
(nichtlineare Verzerrung der Zeitachsen von $\mathbf{X}$ und $\mathbf{Y}$)

Kriterium: Zeitliche Anpassung so, dass Gesamtdistanz $D(\mathbf{X}, \mathbf{Y})$ minimal

Vorgehen: Bestimmung der sogenannten Warping-Kurve
(Abbildung der Zeitachsen von $\mathbf{X} \leftrightarrow \mathbf{Y}$)

Dynamische Zeitanpassung

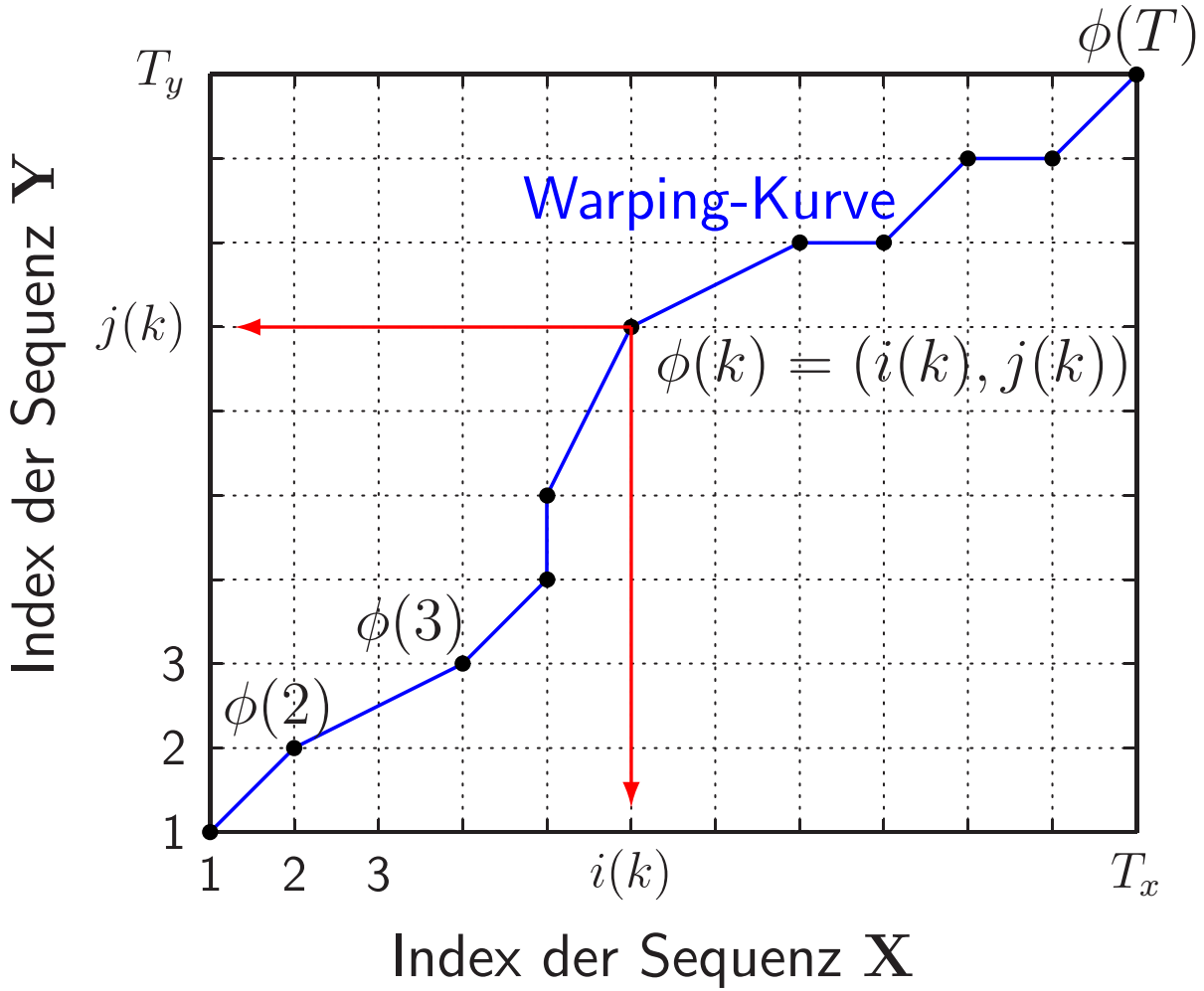
Dynamic Time Warping (DTW)



unangepasst

angepasst

Warping-Kurve



Distanz entlang der Warping-Kurve

Lokale Distanzen auf der Warping-Kurve:

$$d(\phi(k)) \;=\; d(\mathbf{x}_{i(k)}, \mathbf{y}_{j(k)})$$

Gesamtdistanz entlang der Warping-Kurve:

$$D_\phi(\mathbf{X}, \mathbf{Y}) \;=\; \sum_{k=1}^{T} d(\phi(k))\; w(k)$$

Bedingungen für die Warping-Kurve $\phi(k) = (i(k), j(k))$

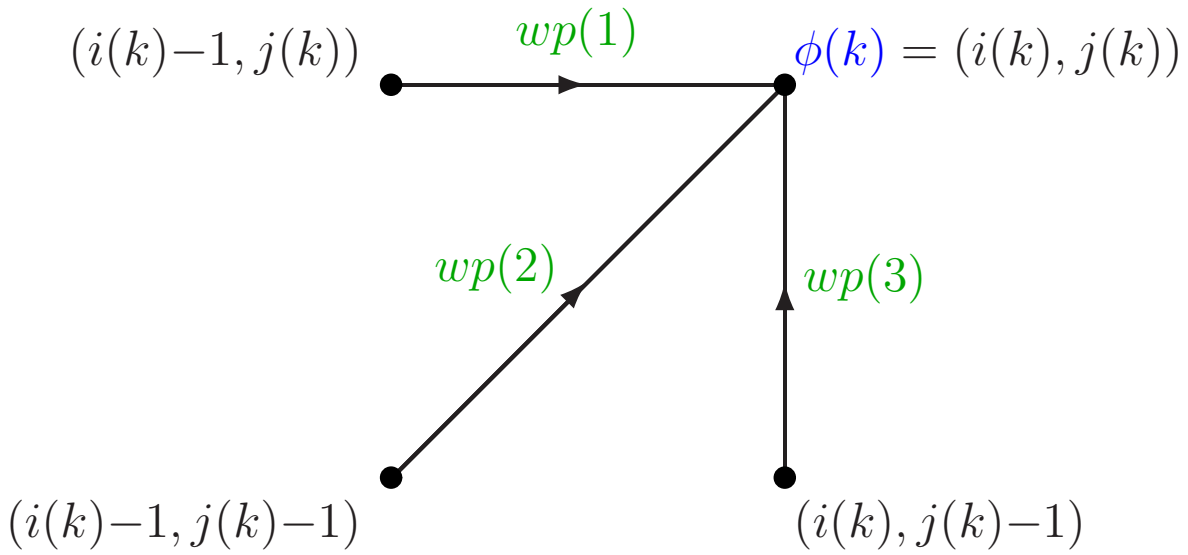
Monotonie: $$\begin{aligned} i(k) &\geq i(k-1) \\ j(k) &\geq j(k-1) \end{aligned}$$ >>>

Lokale Kontinuität: $$\begin{aligned} |i(k) - i(k-1)| &\leq g_x \\ |j(k) - j(k-1)| &\leq g_y \end{aligned}$$ >>>

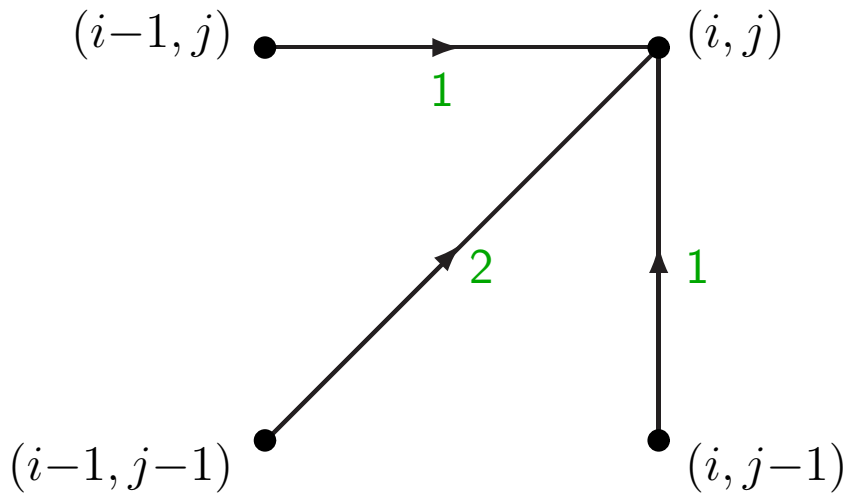
Anfangs- und Endpunkt: $$\begin{aligned} \phi(1) &= (1,1) \\ \phi(T) &= (T_x, T_y) \end{aligned}$$ >>>

Pfaderweiterungen

$$\phi(k-1) \in \left\{ \begin{array}{rcl} \phi(k) - p(1) & = & \phi(k) - (1,0) \\ \phi(k) - p(2) & = & \phi(k) - (1,1) \\ \phi(k) - p(3) & = & \phi(k) - (0,1) \end{array} \right\}$$



Gewichtung der lokalen Distanzen



horizontal:	$1 \cdot d(i, j)$
diagonal:	$2 \cdot d(i, j)$
vertikal:	$1 \cdot d(i, j)$

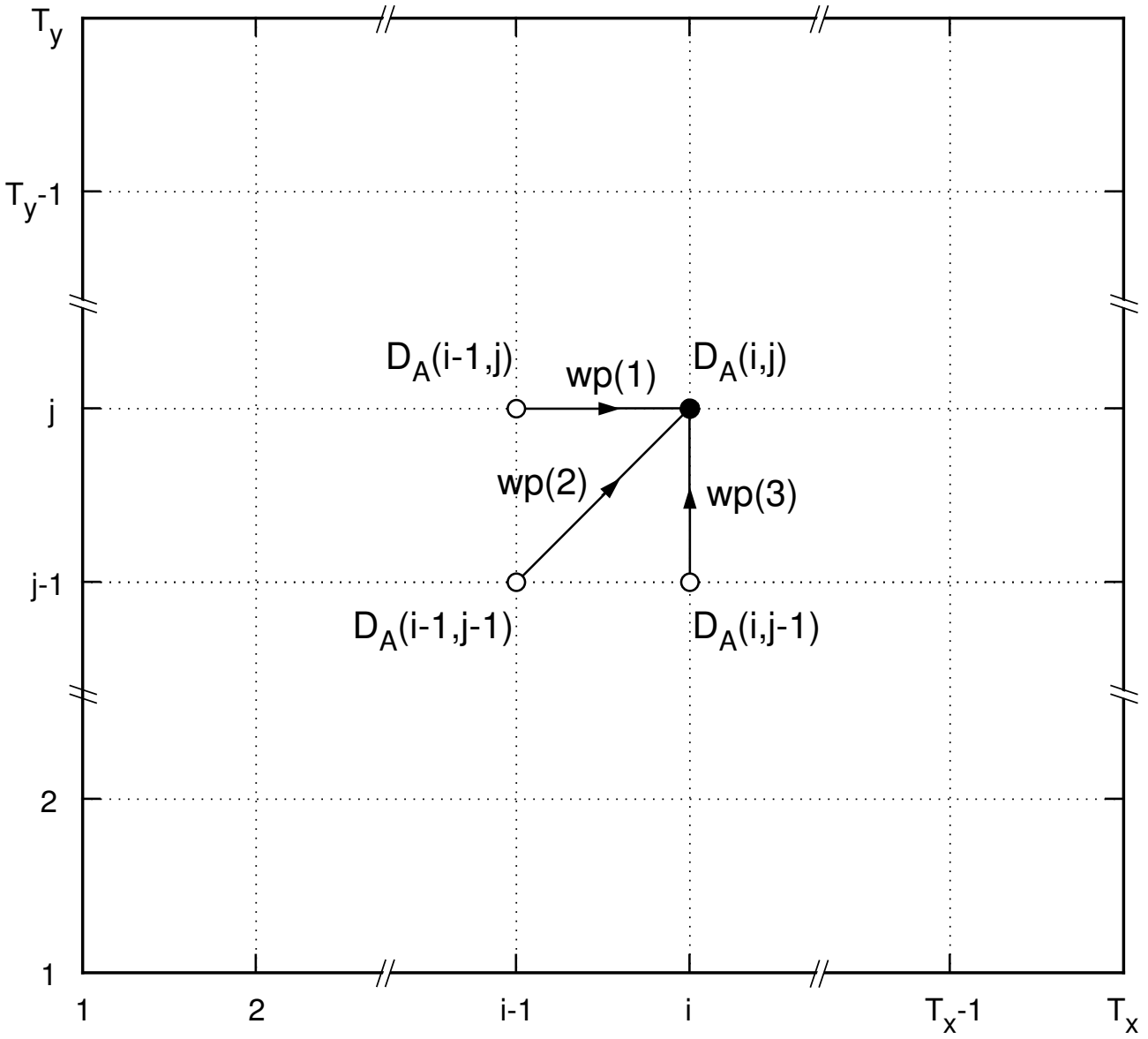
Ermitteln der Warping-Kurve

Gegeben:
- Merkmalssequenzen $\mathbf{X}$ und $\mathbf{Y}$
- Pfaderweiterungen

Gesucht: Warping-Kurve mit minimaler Gesamtdistanz $D_\phi(\mathbf{X}, \mathbf{Y})$

Lösung: Dynamische Programmierung

Die optimale Gesamtlösung kann aus optimalen Teillösungen zusammengesetzt werden.


T_y
T_y-1
j
j-1
2
1
1
2
i-1
i
T_x-1
T_x
D_A(i-1,j)
wp(1)
D_A(i,j)
wp(2)
wp(3)
D_A(i-1,j-1)
D_A(i,j-1)

DTW-Algorithmus

Initialisierung:
- lokale Distanzen $d(i,j)$ mit $1 \leq i \leq T_x$ und $1 \leq j \leq T_y$
- akkumulierte Distanz $D_A(1,1) = d(1,1)$

Rekursion: Ermitteln von akkumulierten Distanzen, bis $D_A(T_x, T_y)$ vorliegt:
- Wahl eines Punktes (i,j), wobei D_A der Vorgänger bekannt
  ($D_A(i,j) = \infty$ für $i < 1$ oder $j < 1$)
- Bestimmung von $D_A(i,j)$ >>>
- Index m der Pfaderweiterung in $\Psi(i,j)$ speichern

Backtracking: Ermitteln der Warping-Kurve $\phi(k),\ k = 1 \ldots T$ aus Ψ
- Endpunkt: $\phi(T) = (T_x, T_y)$
- Vorgängerpunkte: $\phi(k-1) = \phi(k) - p(\Psi(\phi(k)))$

DTW-Demo >>>

Spracherkennung mittels Mustervergleich

Vergleiche das Testmuster eines unbekannten Wortes

$$\mathbf{X} = \mathbf{x}_1 \mathbf{x}_2 \ldots \mathbf{x}_{T_X}$$

mit den gegebenen Referenzmustern eines Vokabulars V

$$\mathbf{Y}^{(j)} = \mathbf{y}_1^{(j)} \mathbf{y}_2^{(j)} \ldots \mathbf{y}_{T_{Y_j}}^{(j)}, \qquad 1 \leq j \leq |V|$$

Als erkannt gilt das Wort $v_i \in V$ mit $i = \underset{j=1,\ldots,|V|}{\operatorname{argmin}} D_\phi(\mathbf{X}, \mathbf{Y}^{(j)})$

Gesamtdistanz $D_\phi(T_x, T_y)$

Feststellung: Gesamtdistanz $D_\phi(T_x, T_y)$ nimmt mit der Länge der Muster zu

Konsequenz: Mit $D_\phi(T_x, T_y)$ als Kriterium werden in der Spracherkennung kürzere Wörter systematisch bevorzugt!

Abhilfe: Normierung so, dass Distanz von T unabhängig

- Ermitteln der Summe der Gewichte entlang der Warping-Kurve, wobei $w(k) = wp(\Psi(\phi(k)))$
- $$D_{\phi N}(\mathbf{X}, \mathbf{Y}) = \frac{\sum_{k=1}^{T} d(\phi(k))w(k)}{\sum_{k=1}^{T} w(k)} = \frac{D_A(T_x,T_y)}{\sum_{k=1}^{T} w(k)}$$

Spracherkennung mittels Mustervergleich

Vergleiche das Testmuster eines unbekannten Wortes

$$\mathbf{X} = \mathbf{x}_1 \mathbf{x}_2 \ldots \mathbf{x}_{T_X}$$

mit den gegebenen Referenzmustern eines Vokabulars V

$$\mathbf{Y}^{(j)} = \mathbf{y}_1^{(j)} \mathbf{y}_2^{(j)} \ldots \mathbf{y}_{T_{Y_j}}^{(j)}, \qquad 1 \leq j \leq |V|$$

Als erkannt gilt das Wort $v_i \in V$ mit $i = \underset{j=1,\ldots,|V|}{\operatorname{argmin}} D_{\phi N}(\mathbf{X}, \mathbf{Y}^{(j)})$,

also das Wort mit der kleinsten normierten Gesamtdistanz

Spracherkennung mittels Mustervergleich

Funktioniert nur für den sprecherabhängigen Fall gut!

$\longrightarrow$ Referenzmuster müssen vom Benutzer gesprochen werden!

Ermitteln der Referenzmuster für die Vokabularwörter

Einfacher Ansatz:

- Sprachsignal für Wort $v_i \in V$ aufnehmen
- Merkmalssequenz (MFCC) ermitteln $\longrightarrow$ Referenzmuster $\mathbf{Y}^{(i)}$

Besserer Ansatz: (robuster)

- Sprachsignal für Wort v_i K Mal aufnehmen
- Merkmalssequenzen ermitteln $\longrightarrow$ $\mathbf{X}^{(j)}$, $j = 1 \ldots K$
- alle Distanzen $D_{\phi N}(\mathbf{X}^{(j)}, \mathbf{X}^{(k)})$ mit $j \neq k$ berechnen (DTW)
- Muster $\mathbf{X}^{(r)}$ mit kleinster Distanz zu allen andern bestimmen
- alle Muster $\mathbf{X}^{(j)}$, $j \neq r$ auf $\mathbf{X}^{(r)}$ zeitnormalisieren (DTW)
- normalisierte Muster $\mathbf{X}_n^{(i)}$ mitteln $\longrightarrow$ Referenzmuster $\mathbf{Y}^{(i)}$

>>>

Zusammenfassung: Sprachmustervergleich

- Es braucht eine dynamische zeitliche Anpassung der Sprachmuster
- DTW-Algorithmus: 2 Phasen:
  - Vorwärts: Berechnung der minimalen akkumulierten Distanz
  - Rückwärts: Ermitteln der zugehörigen Warping-Kurve
    (und der Summe der Gewichte für die Normierung)
- Wir müssen spezifizieren:
  - Sprachmerkmale
  - Lokales Distanzmass
  - Bedingungen für die Warping-Kurve

Thema der nächsten Lektion

Einführung in die statistische Spracherkennung

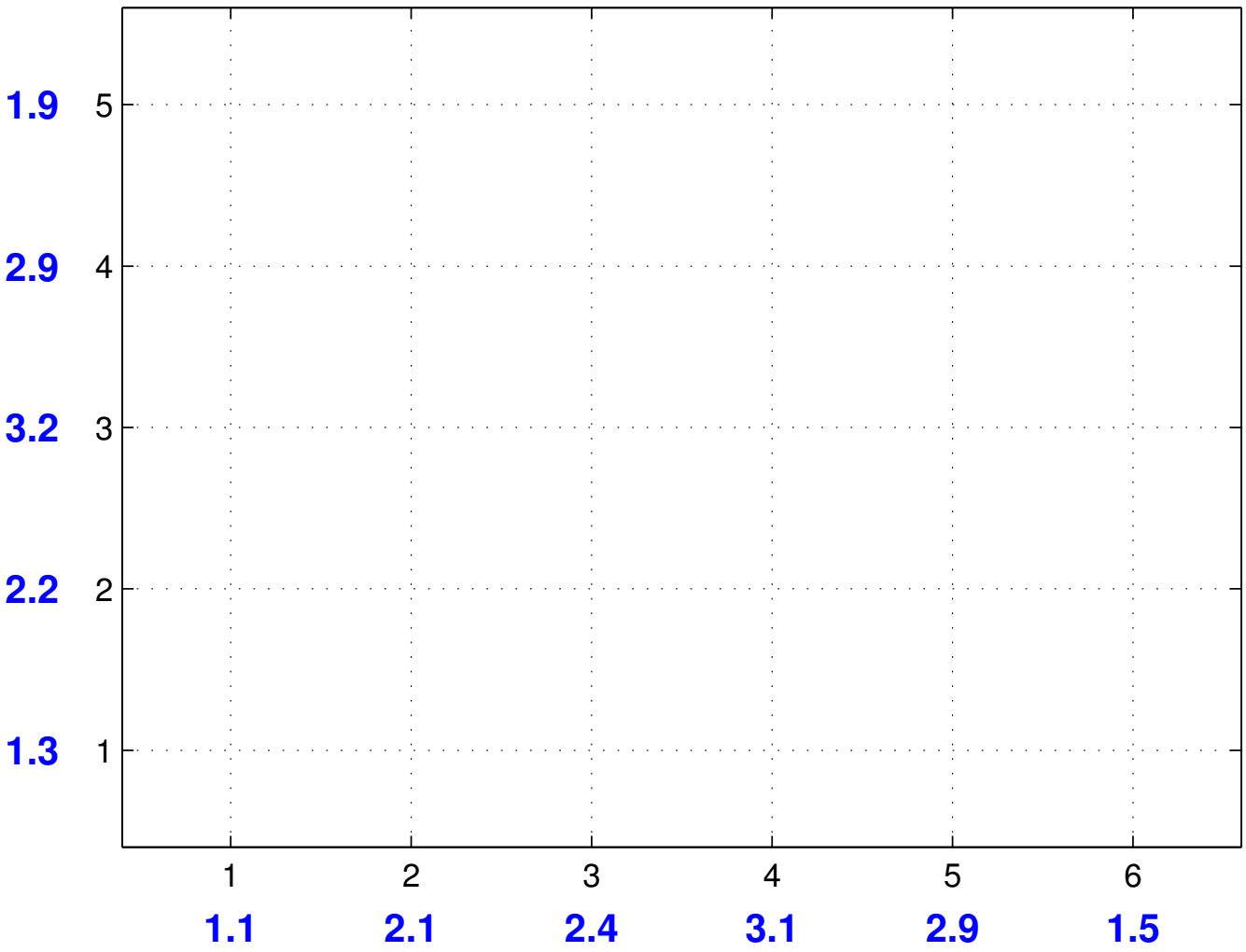
Zur Übersicht der Vorlesung *Sprachverarbeitung I* >>>

Demonstration des DTW-Algorithmus

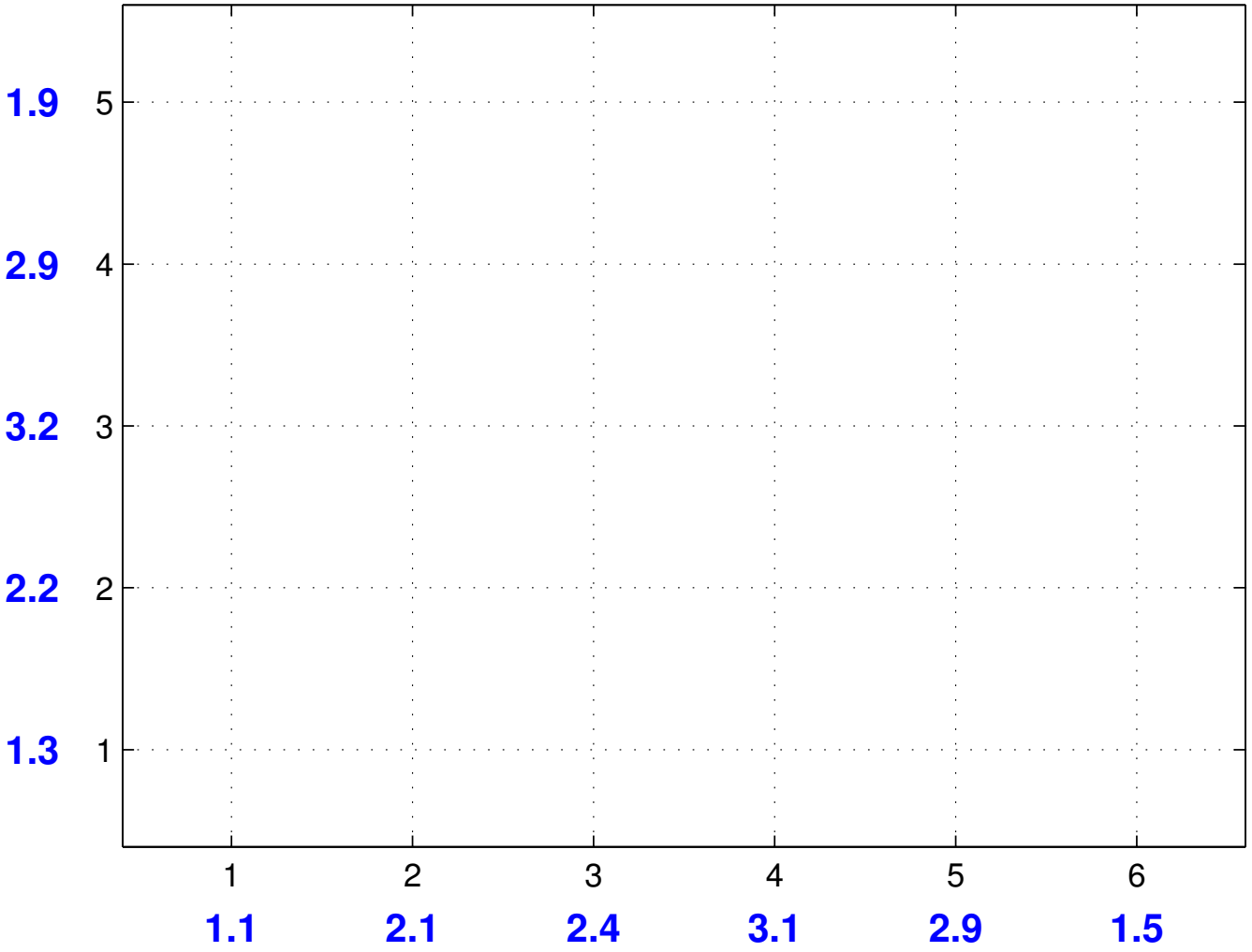
Dynamiche Zeitanpassung zwischen zwei Sequenzen

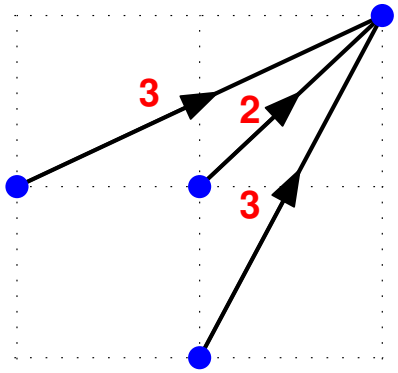
x-Sequenz: 1.1 2.1 2.4 3.1 2.9 1.5

y-Sequenz: 1.3 2.2 3.2 2.9 1.9

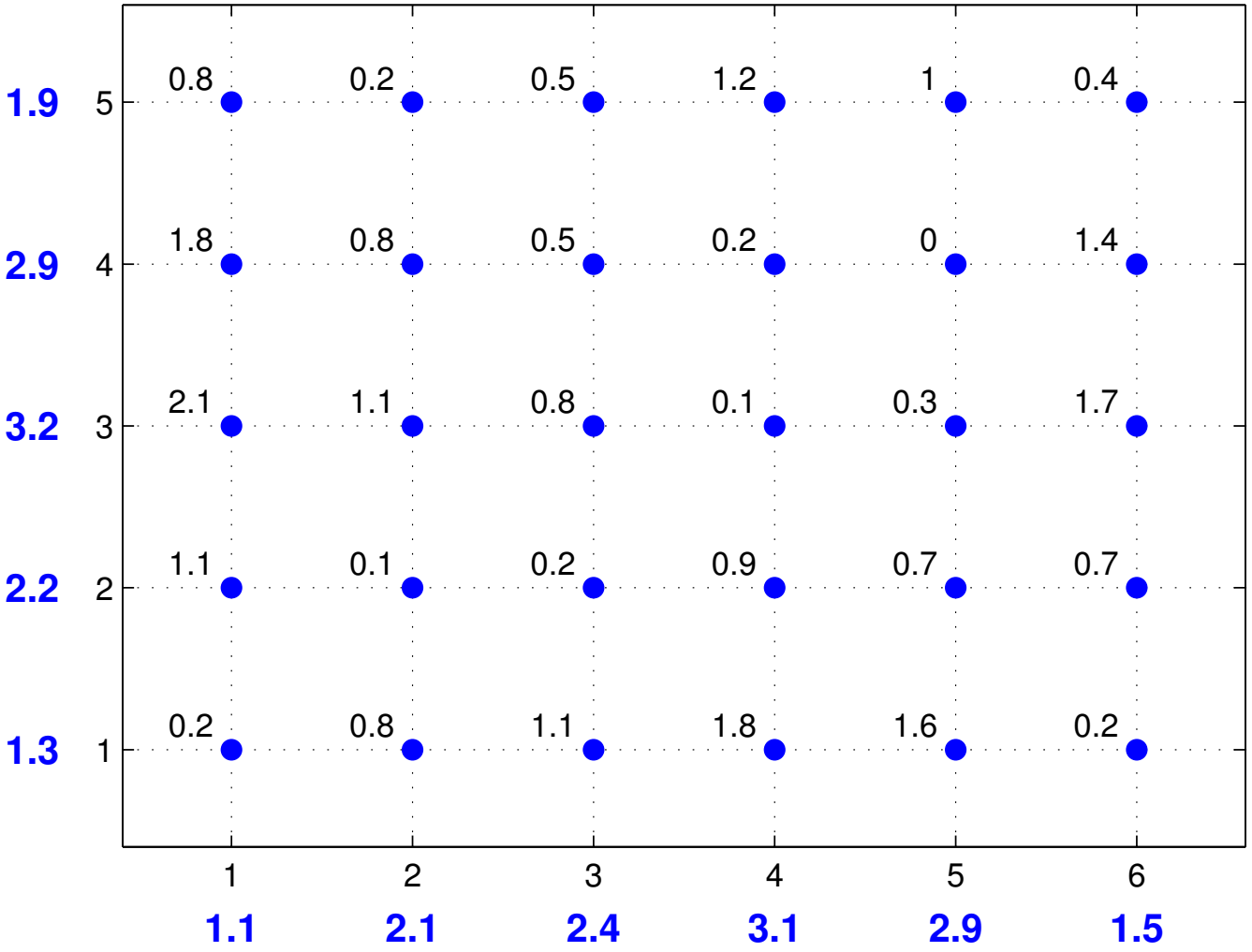

1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
1
2
3
4
5
6
1.1
2.1
2.4
3.1
2.9
1.5

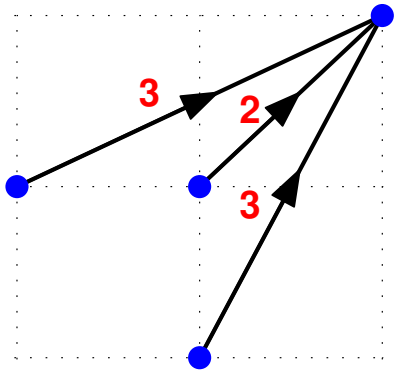
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
1
2
3
4
5
6
1.1
2.1
2.4
3.1
2.9
1.5

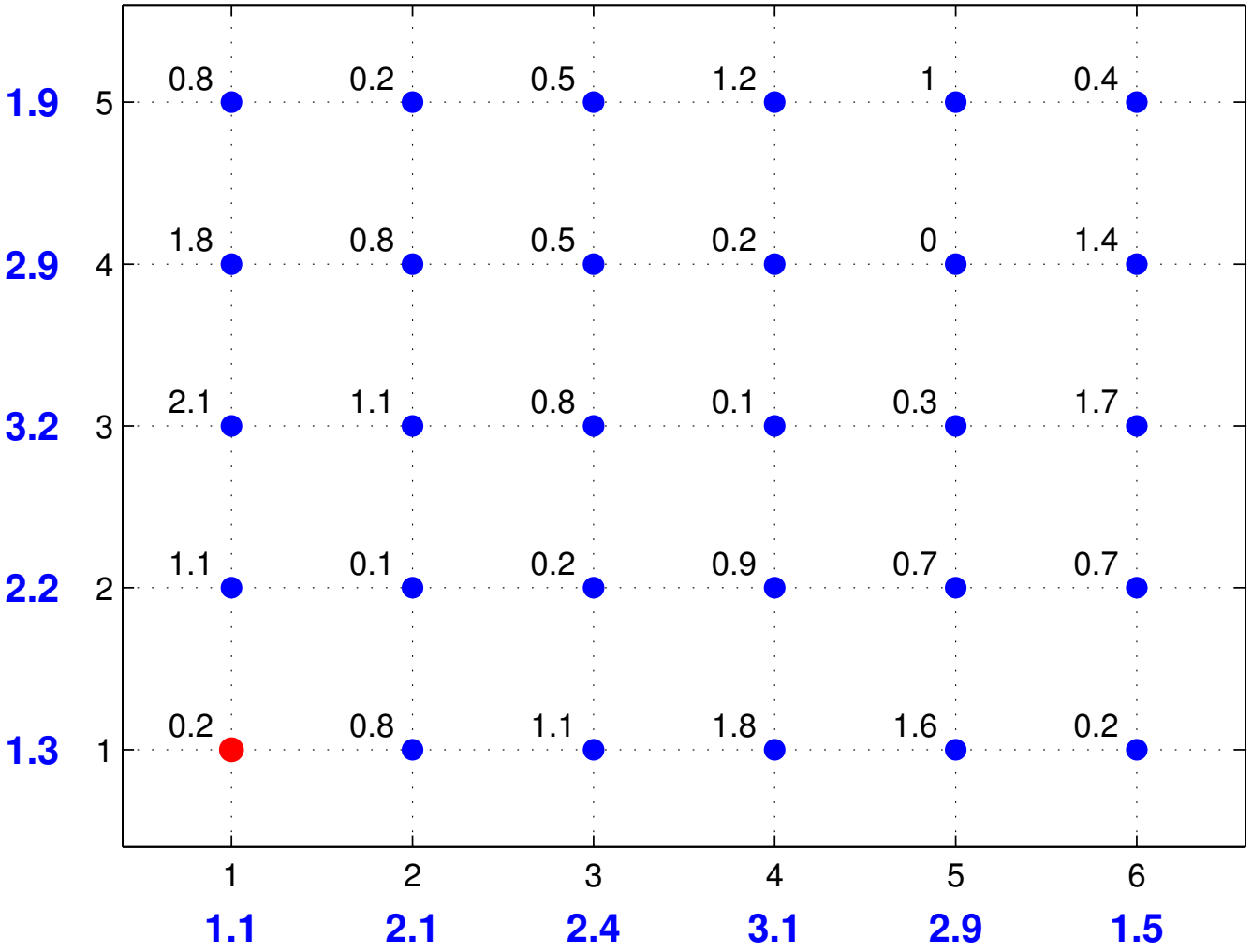

3
2
3

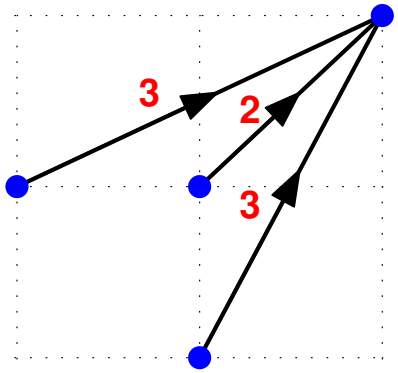
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

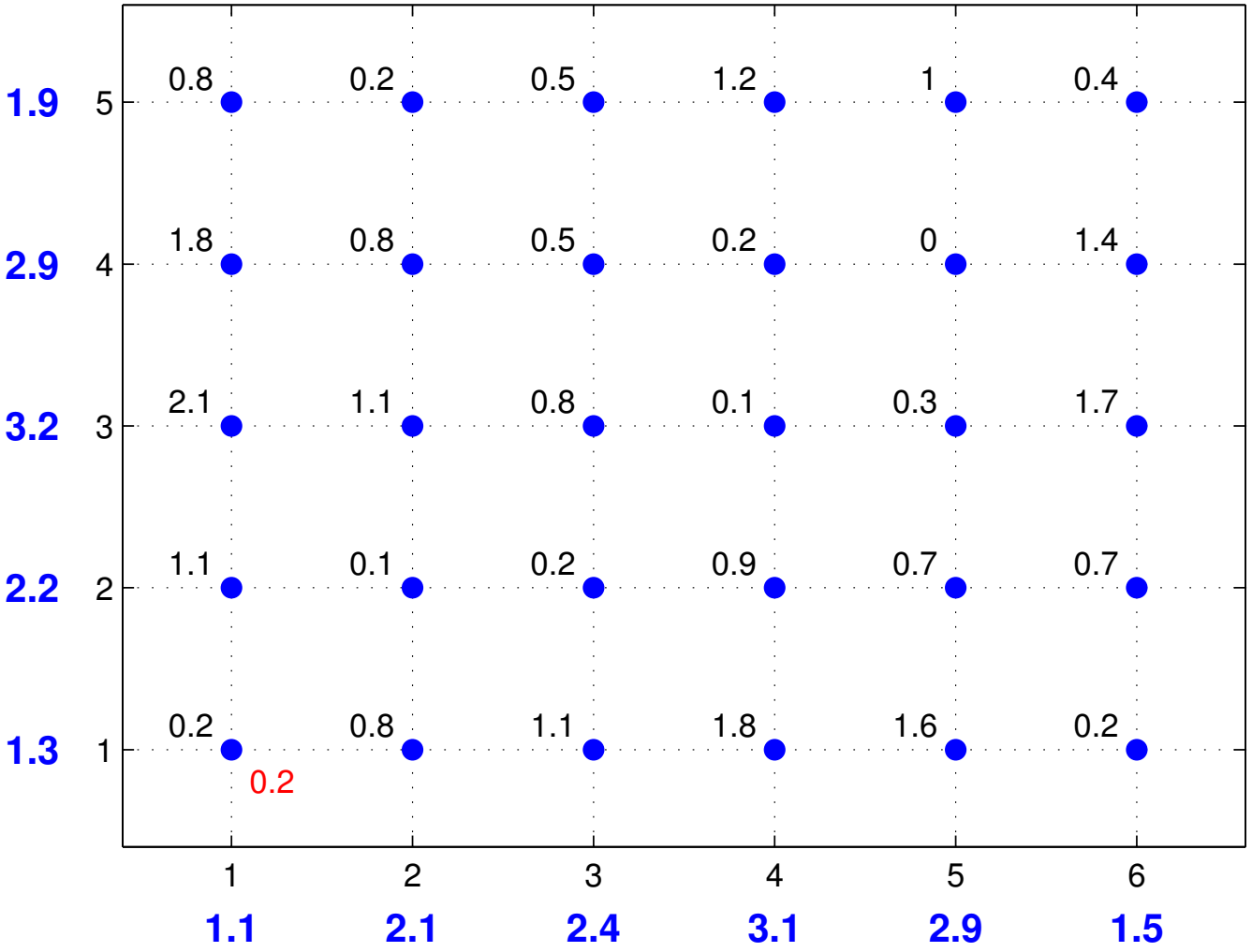

3
2
3

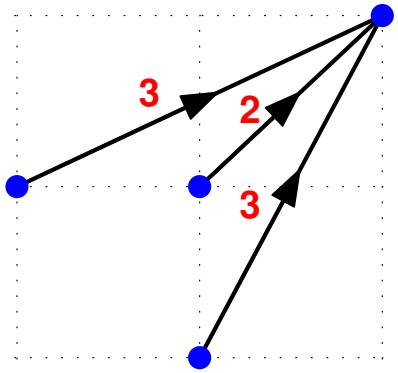
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

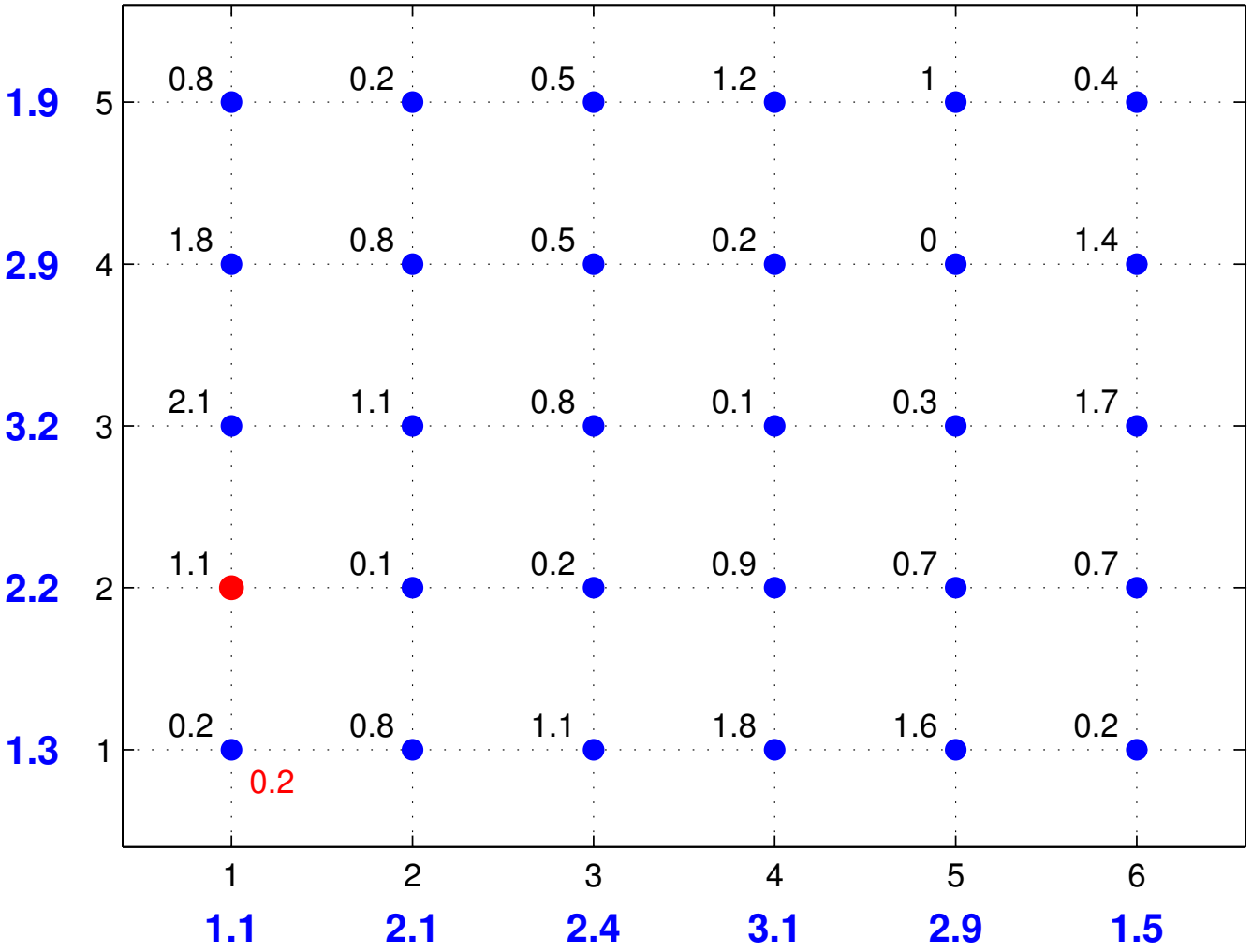

3
2
3

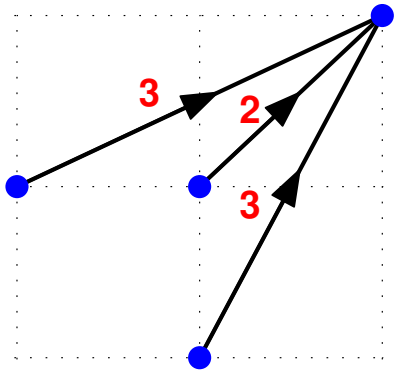
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

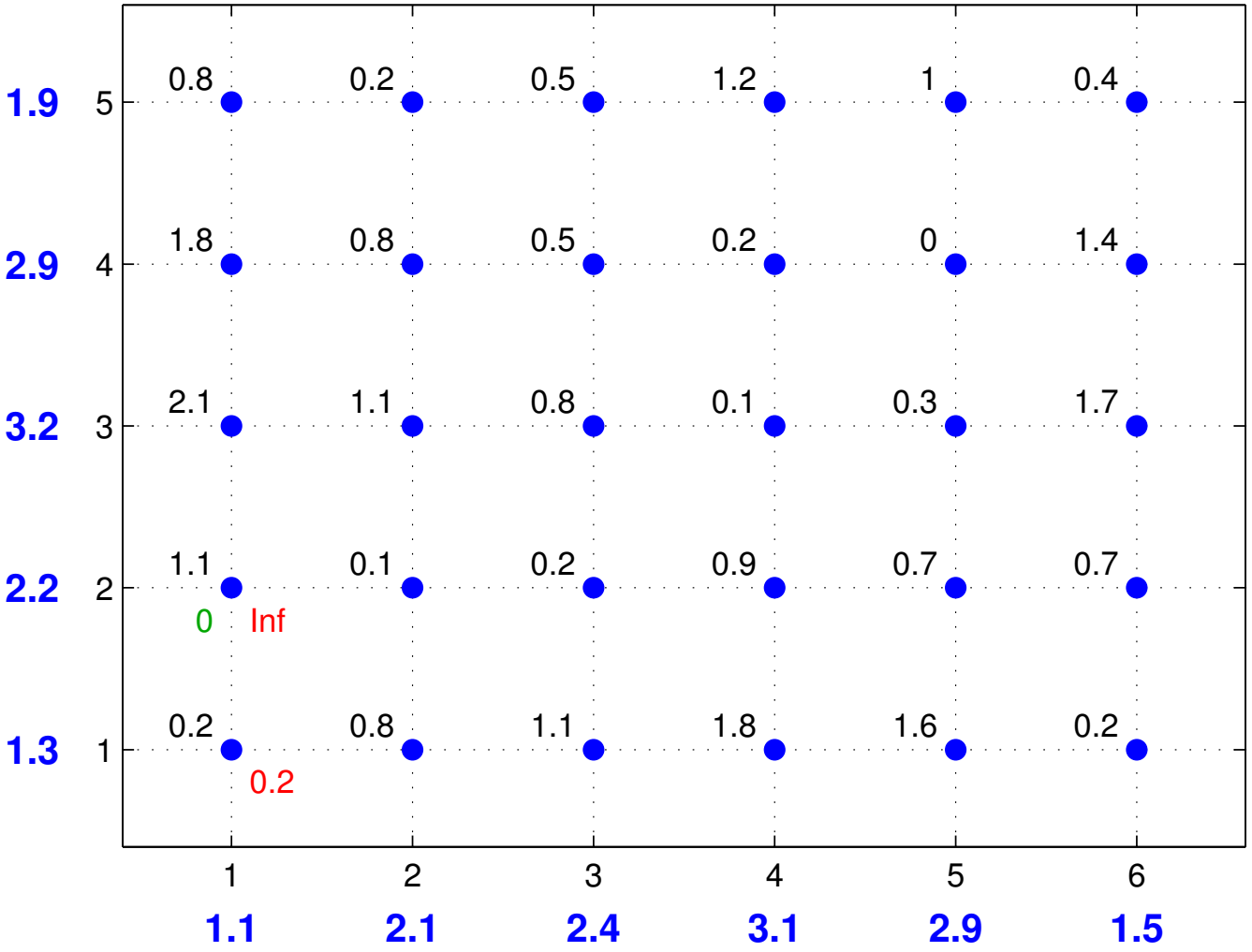

3
2
3

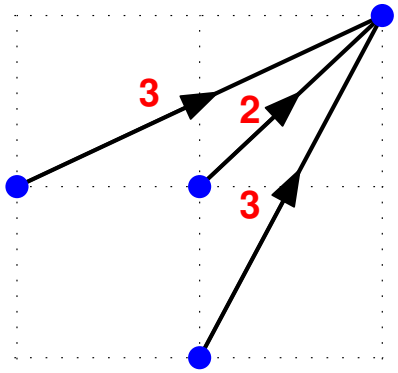
Inf

Inf

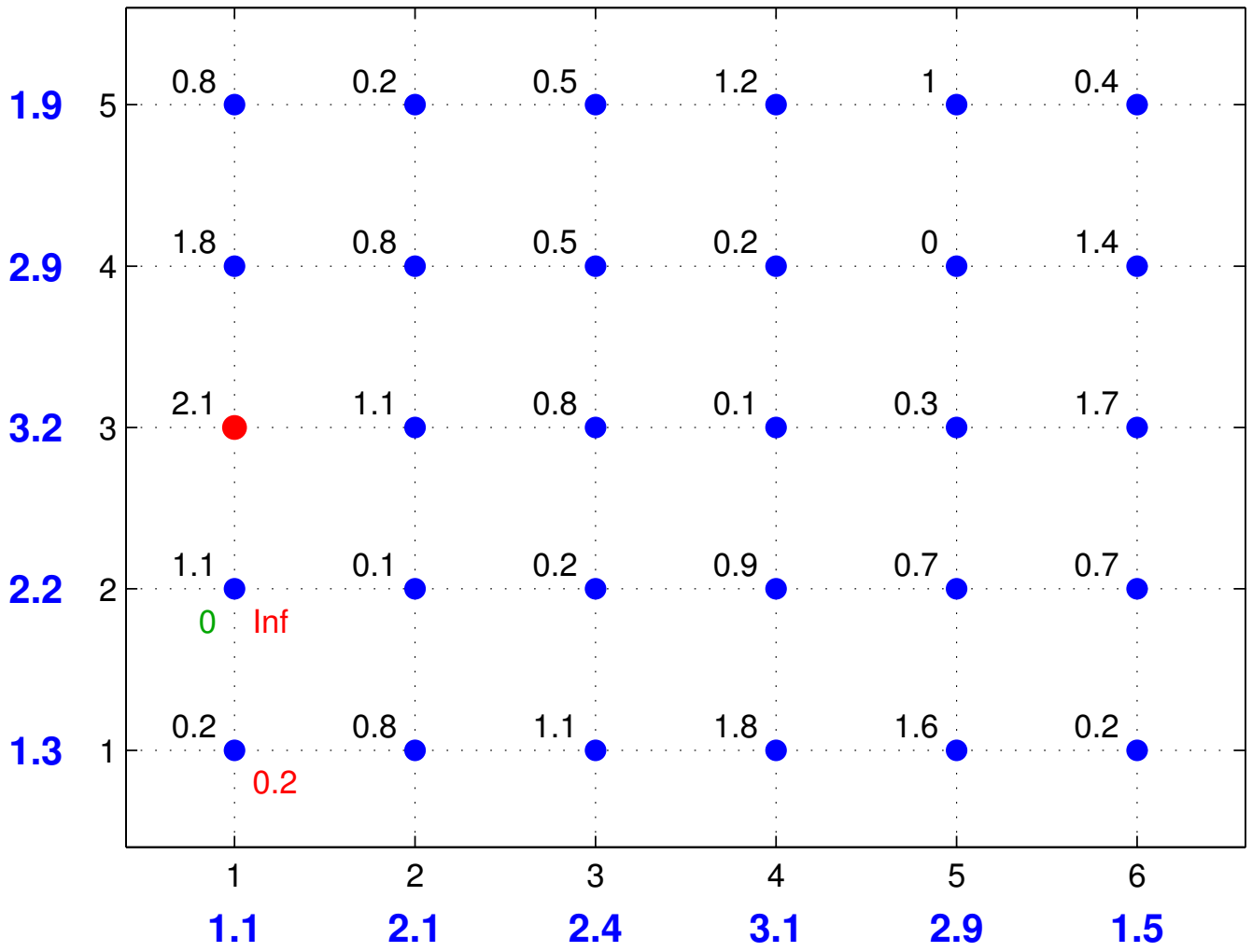
Inf

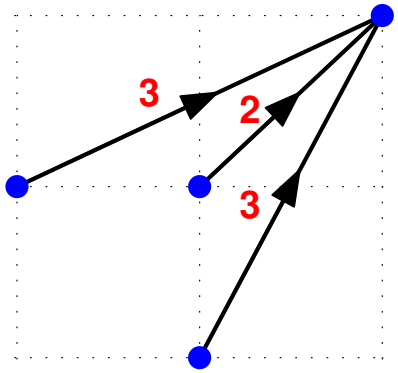
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
0
Inf
0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
0 Inf
0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

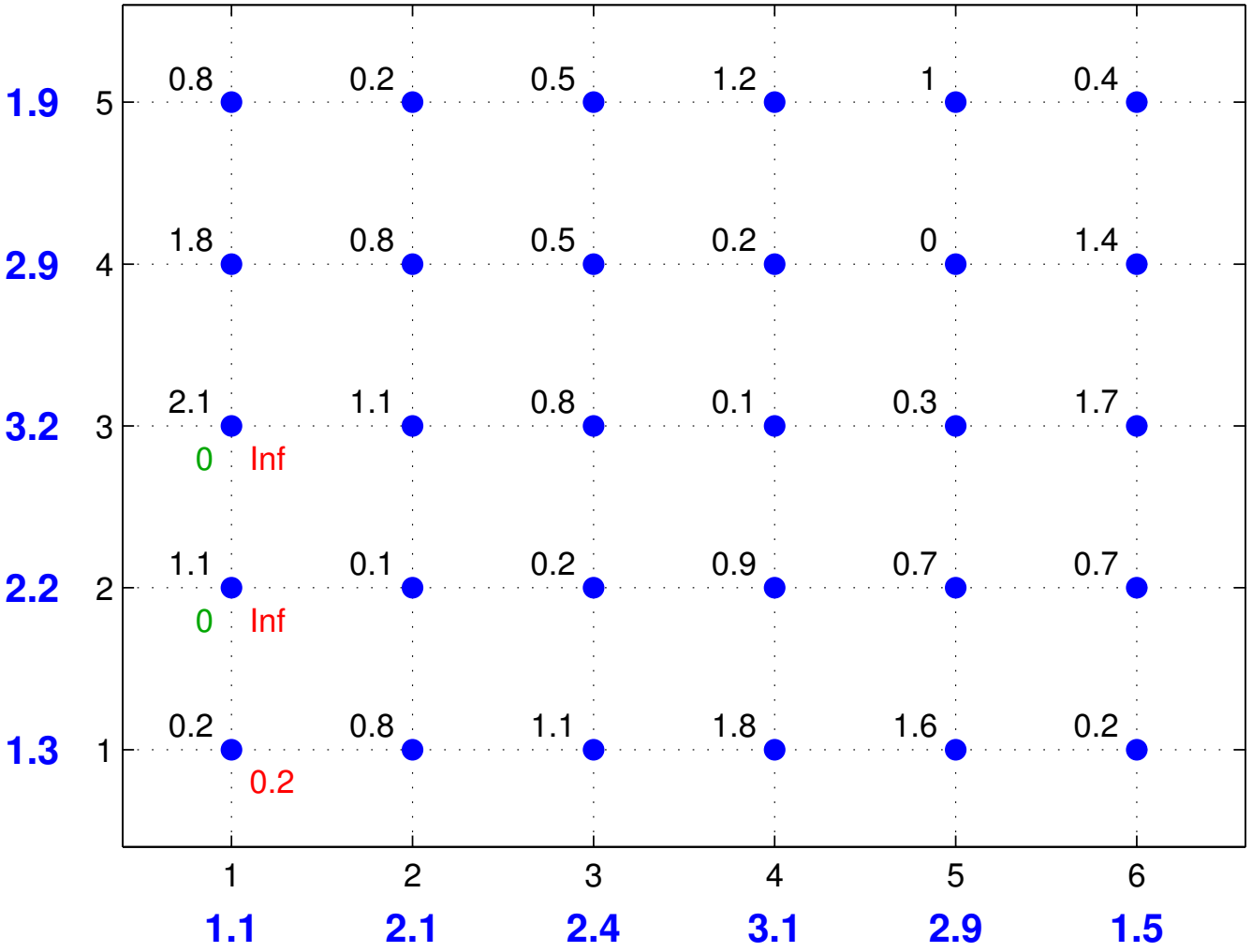

3
2
3

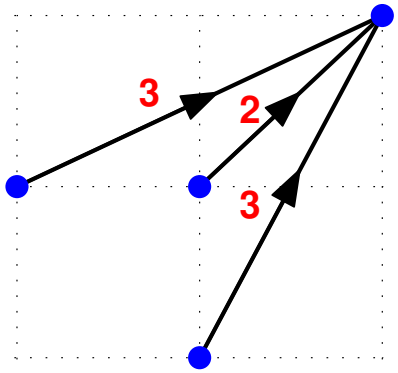
Inf

Inf

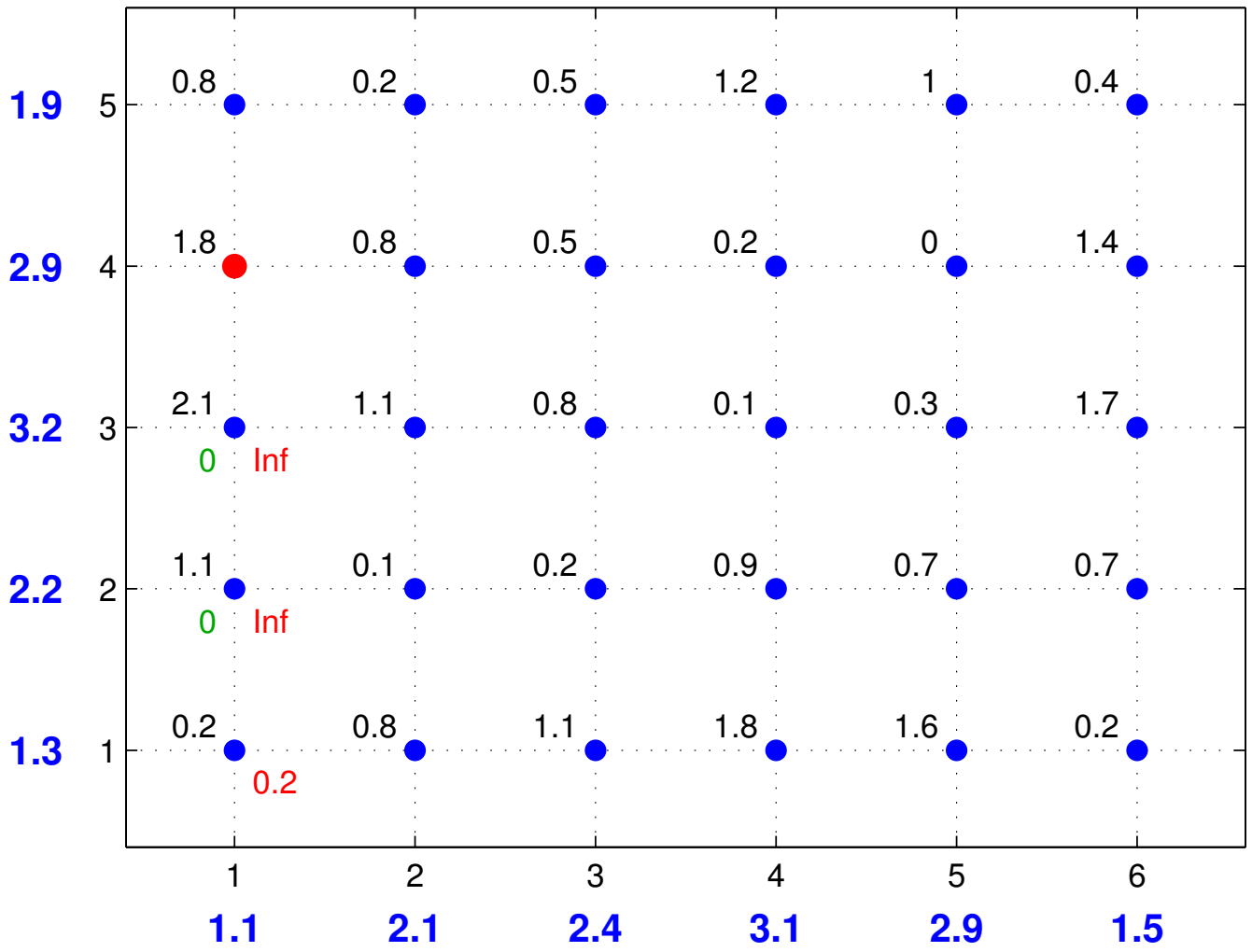
Inf

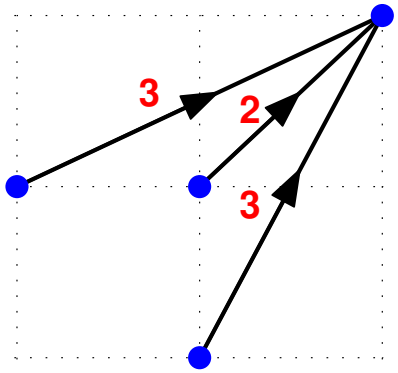
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

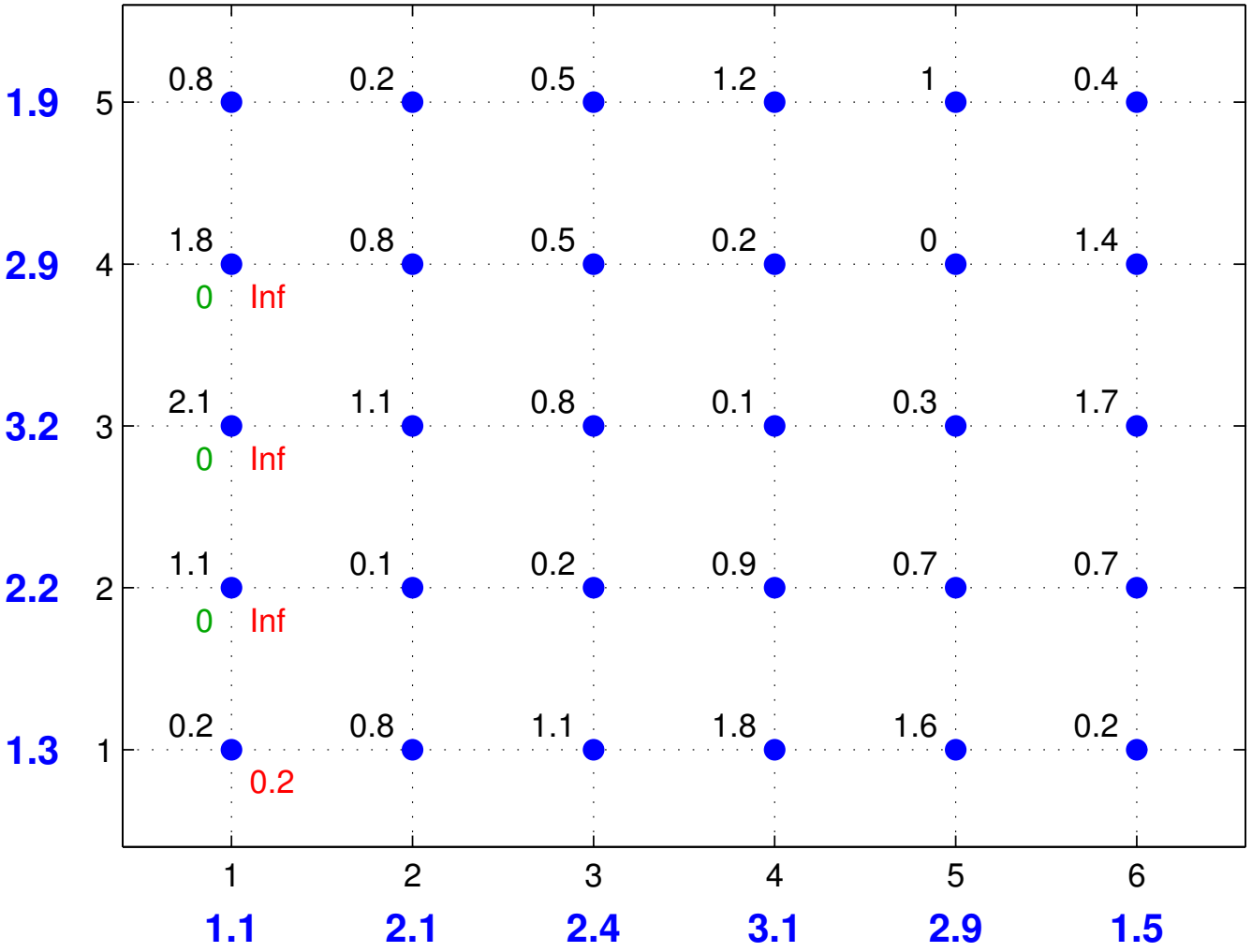

3
2
3

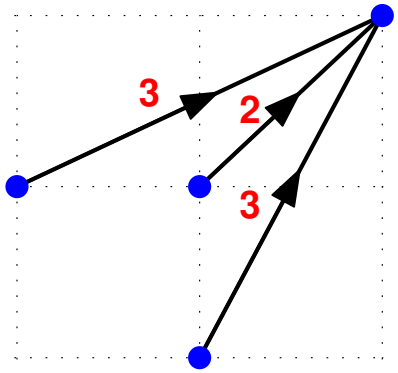
Inf

Inf

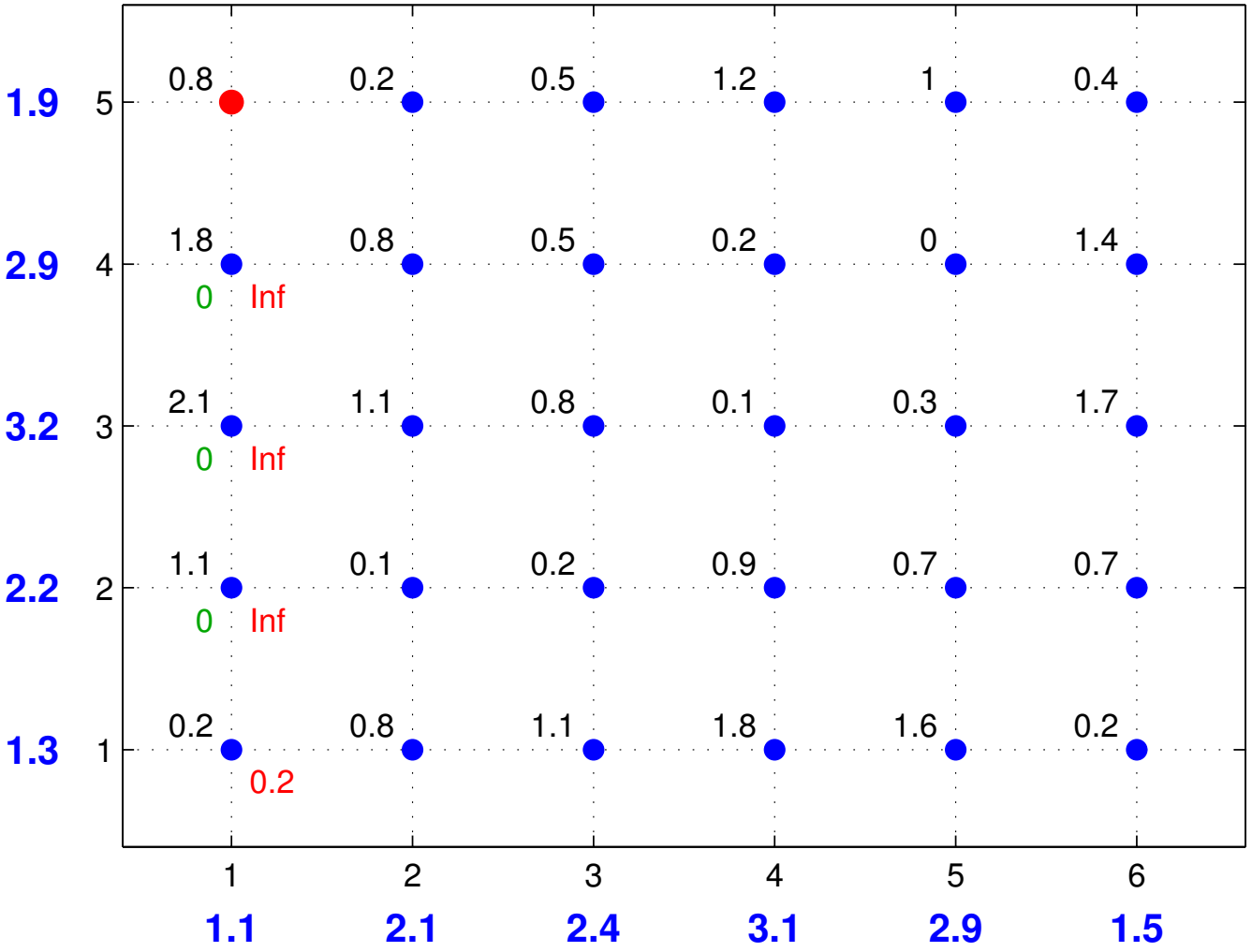
Inf

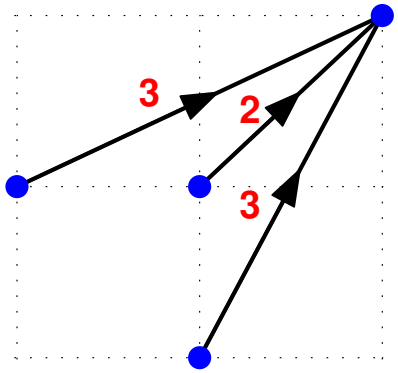
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

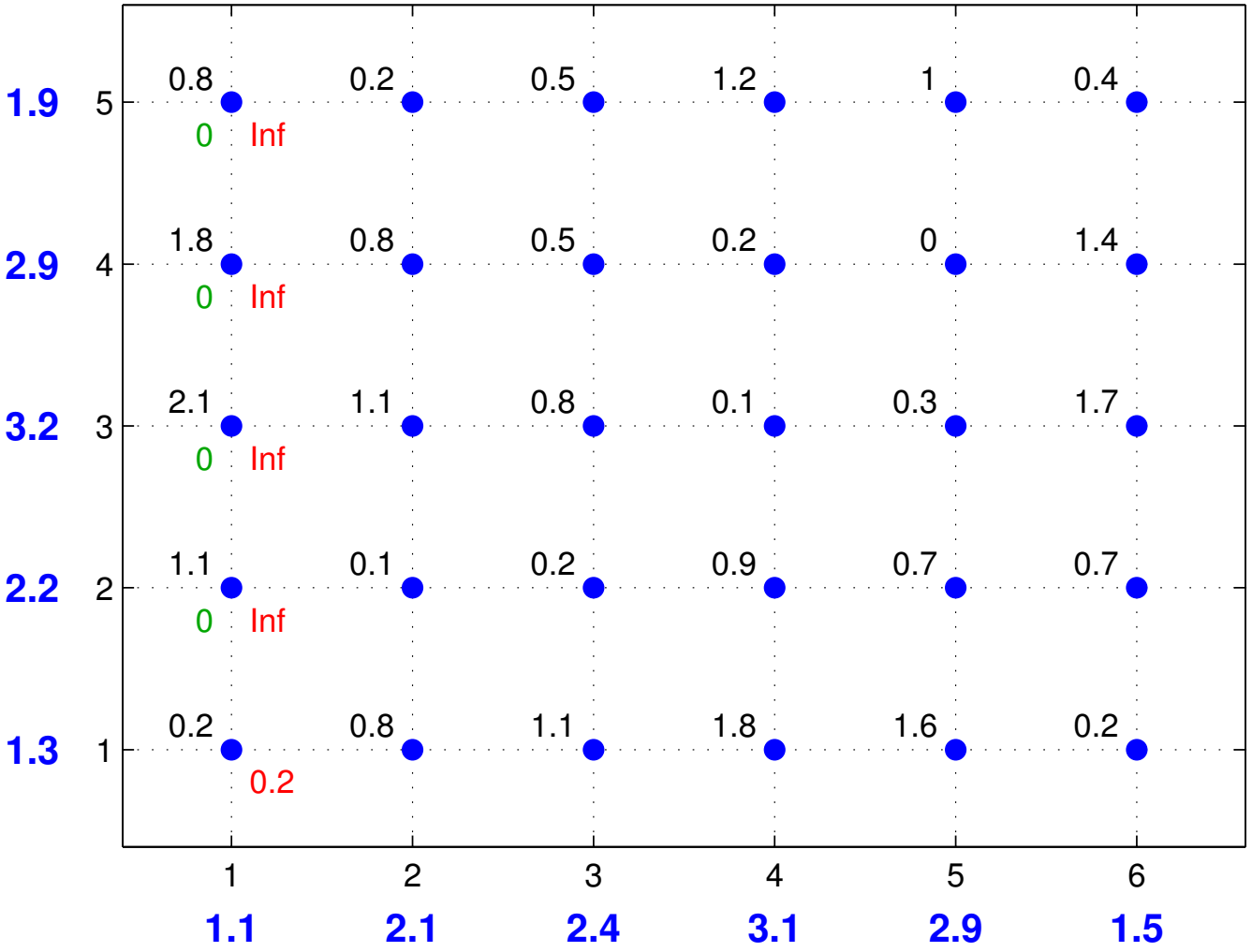

3
2
3

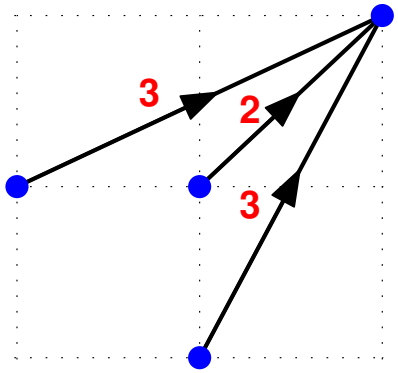
Inf

Inf

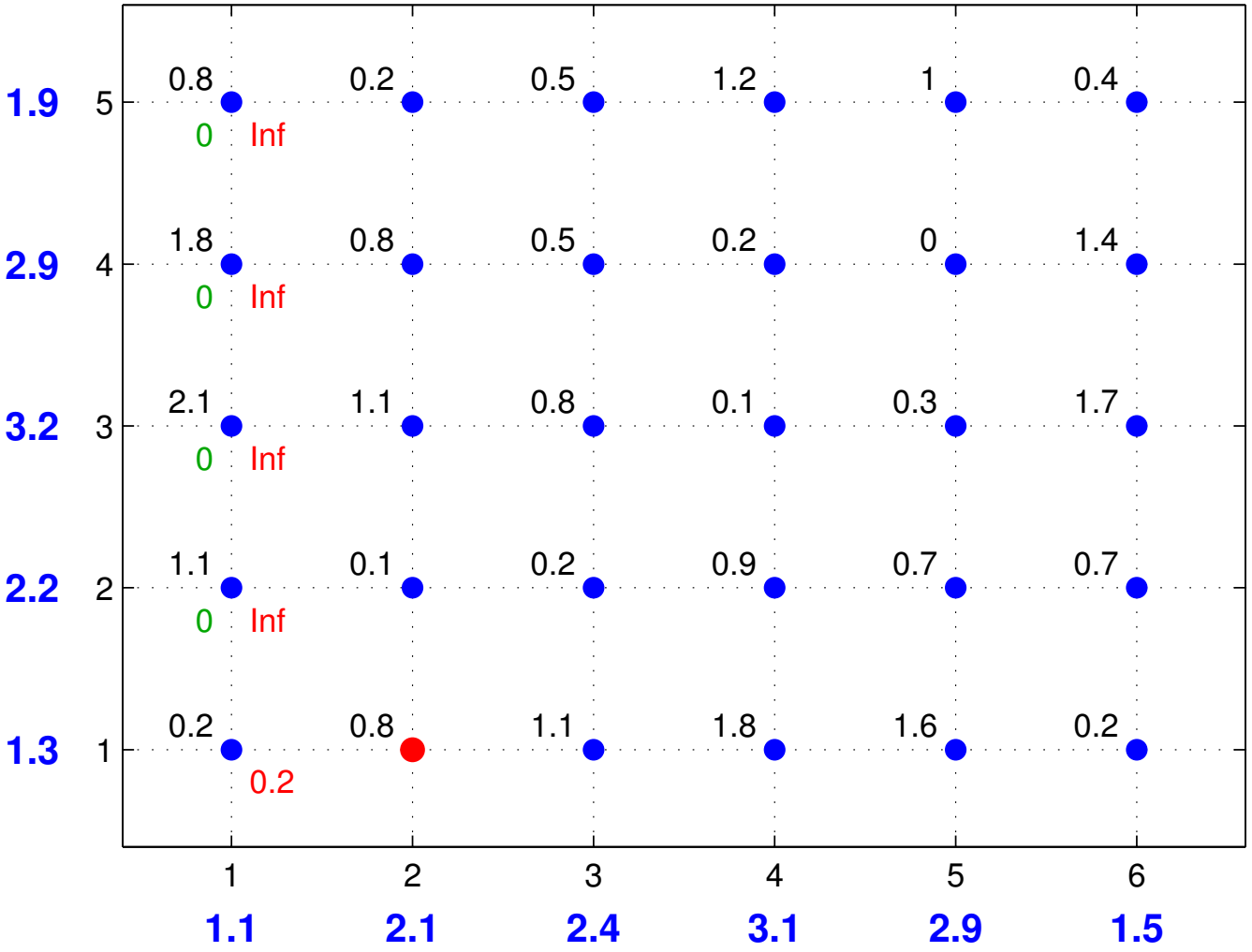
Inf

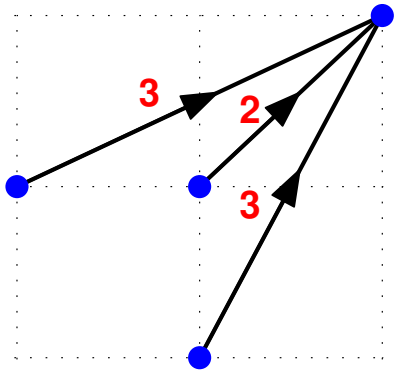
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

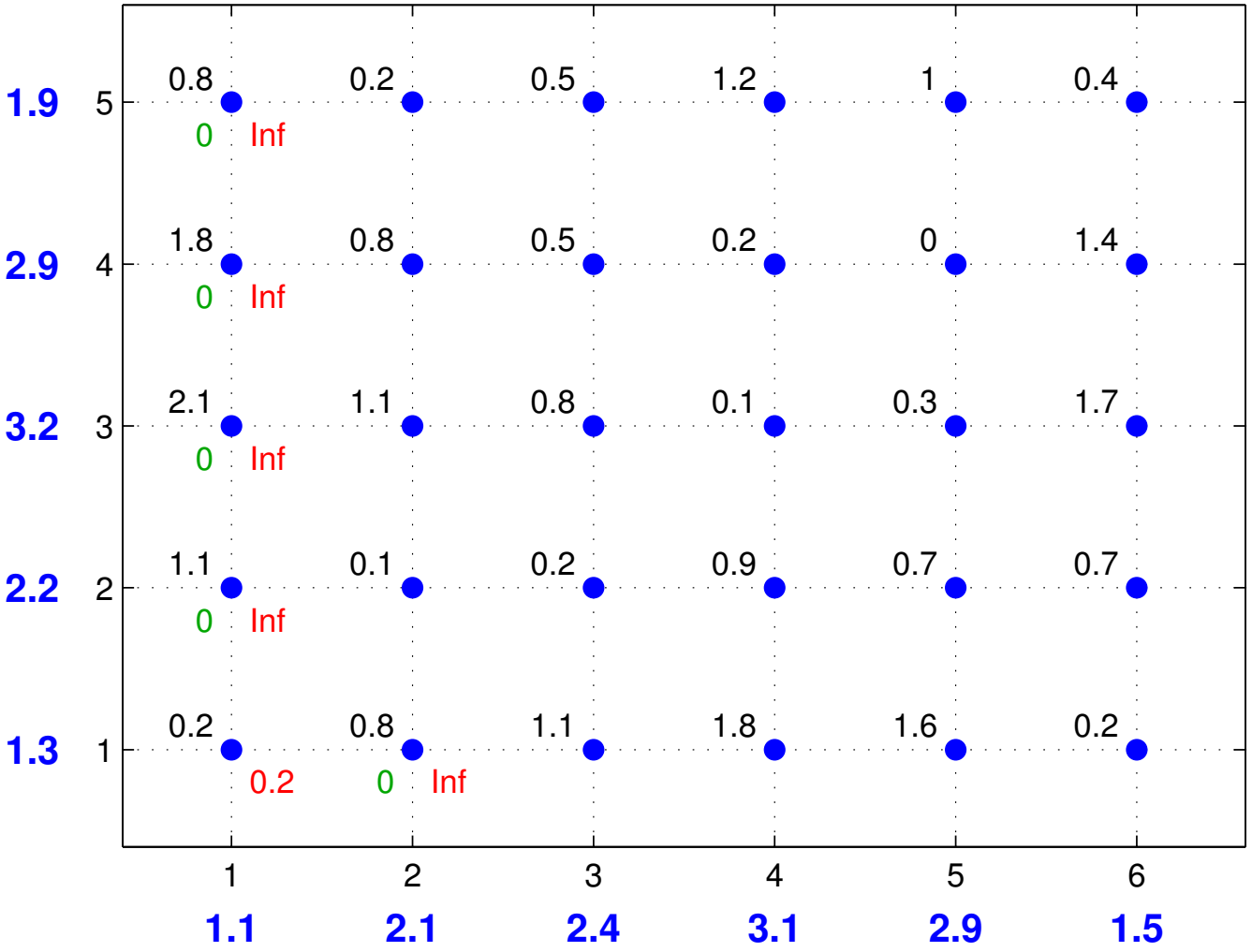

3
2
3

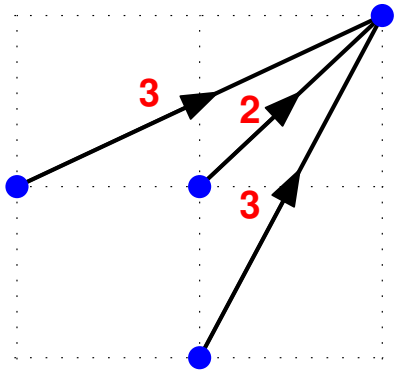
Inf

Inf

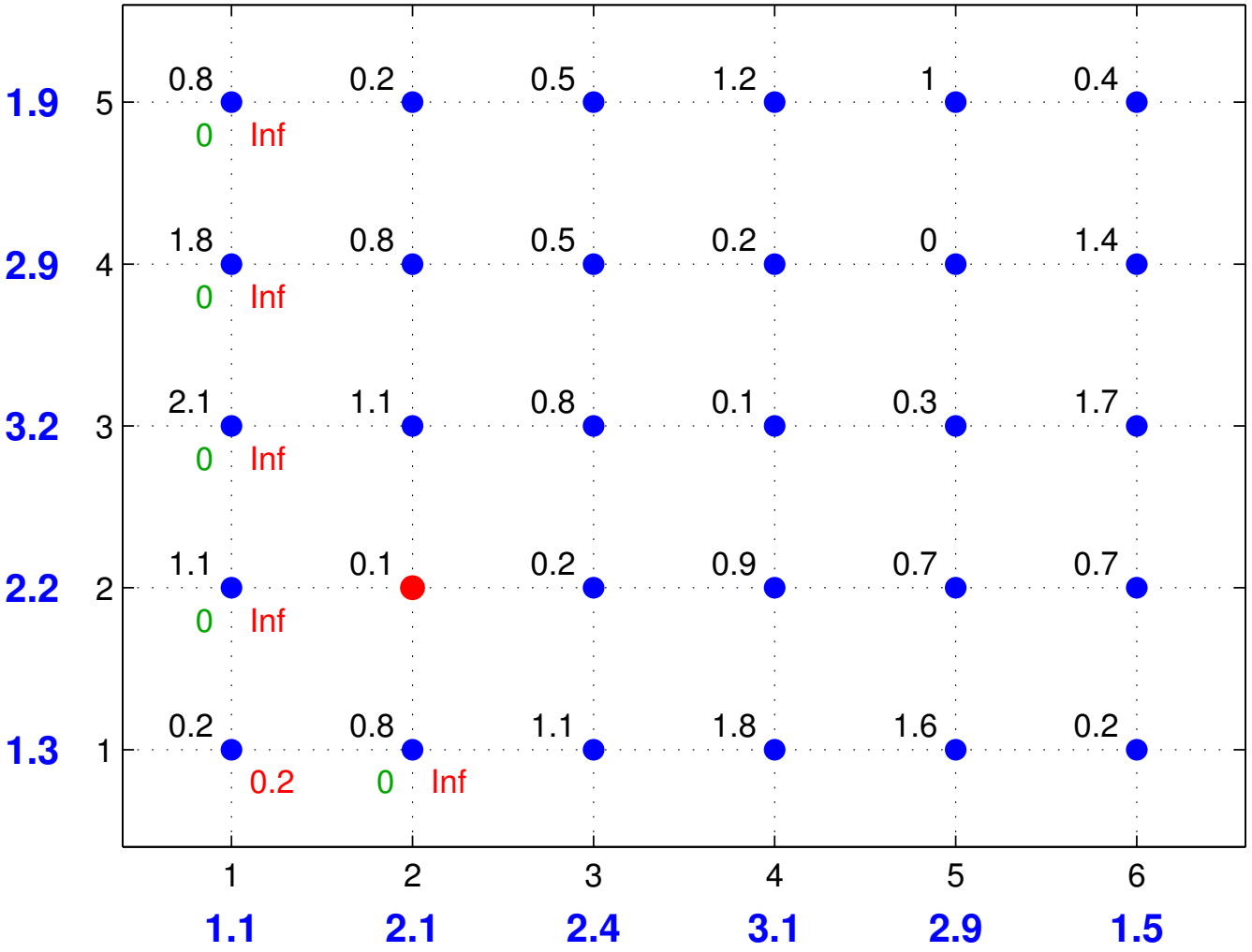
Inf

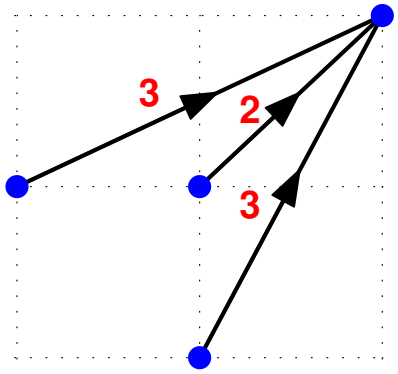
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
0.8 0.2 0.5 1.2 1 0.4
0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

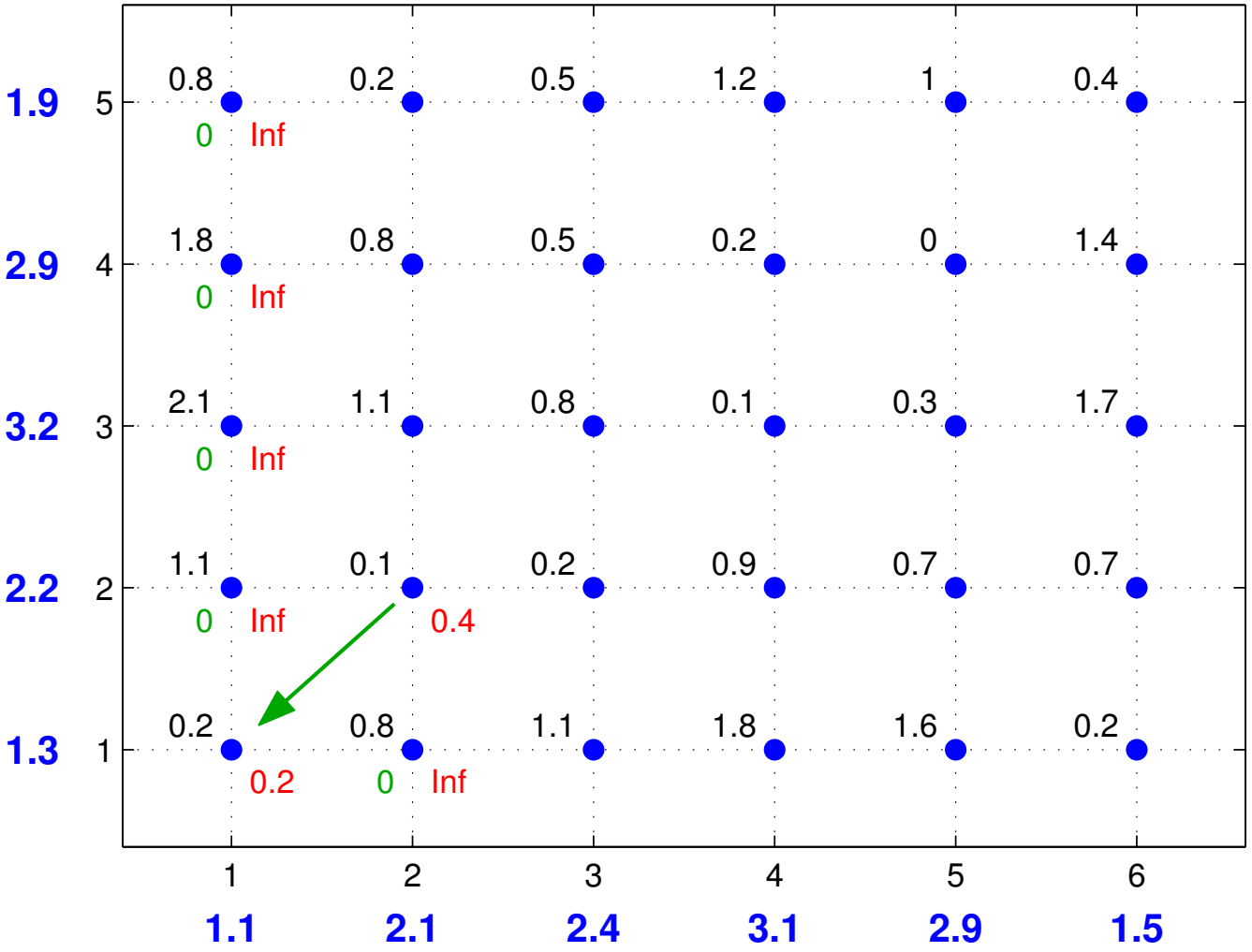

3
2
3

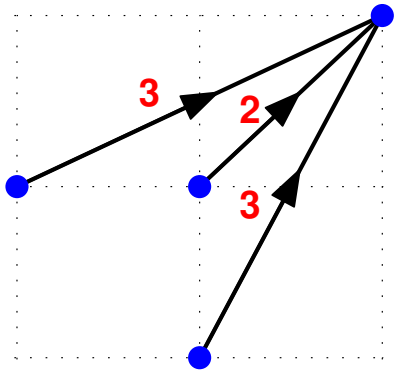
Inf

0.4

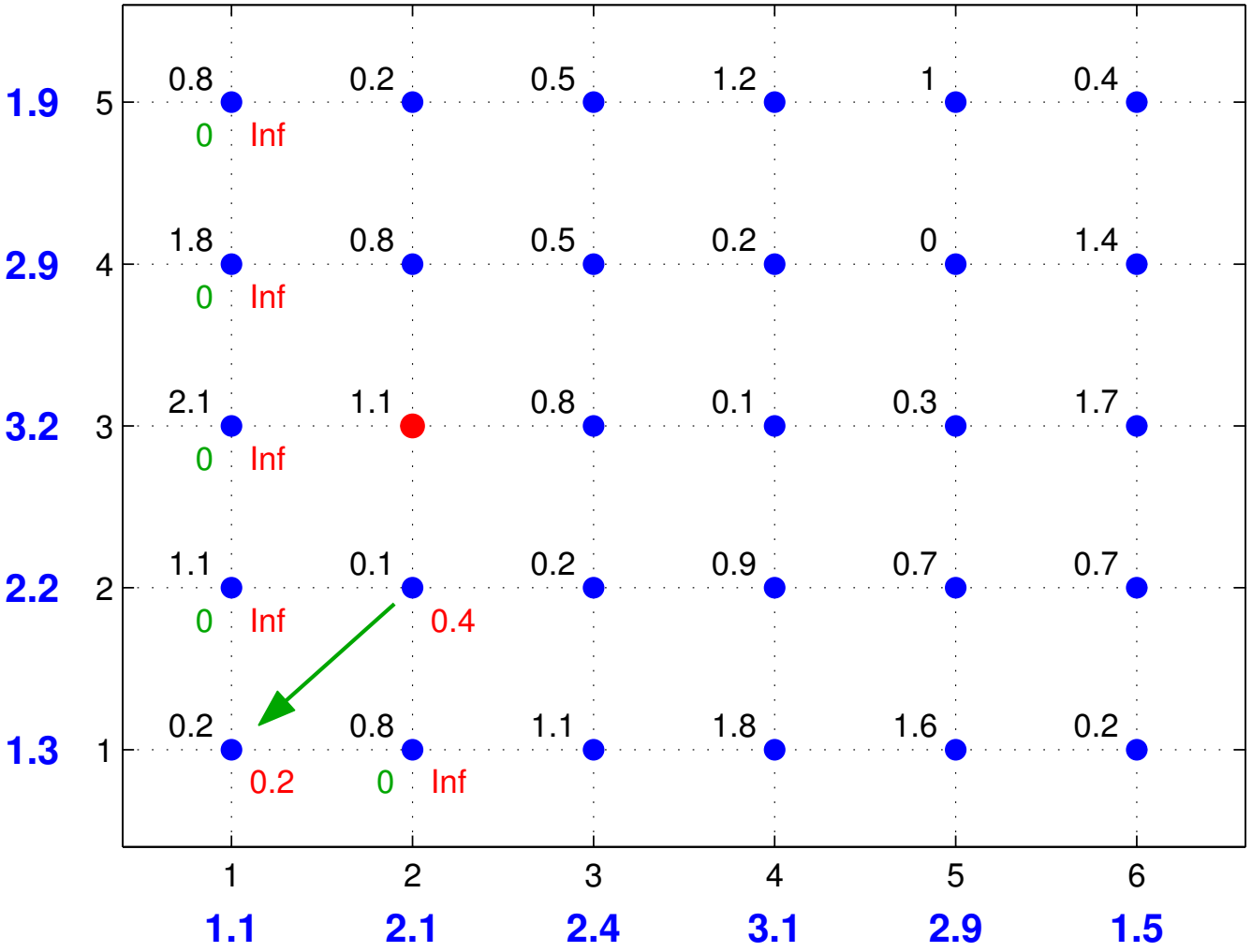
Inf

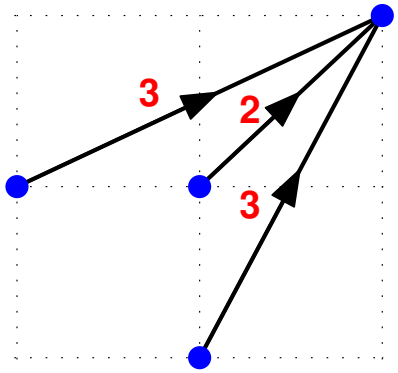
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
0.8 0.2 0.5 1.2 1 0.4
0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

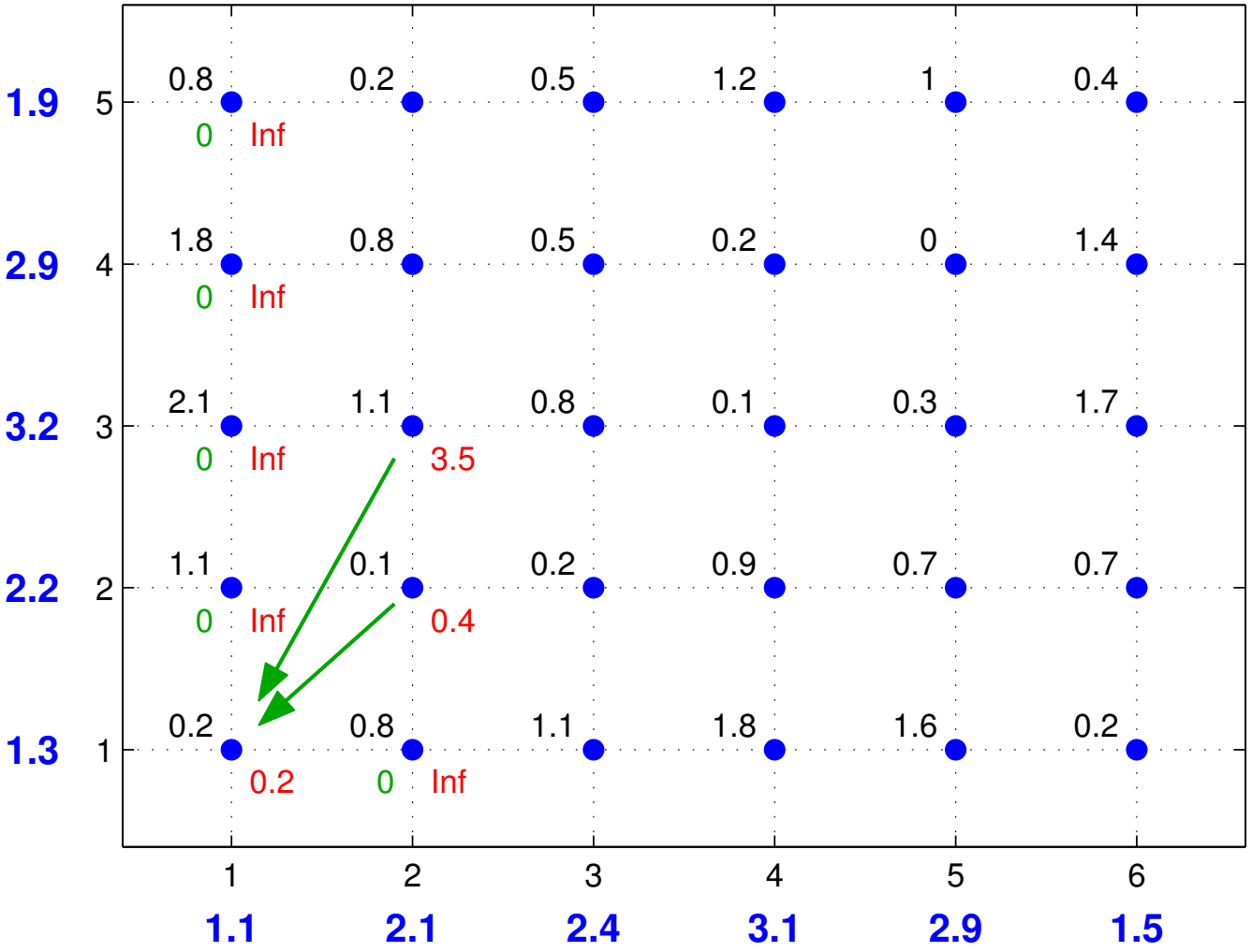

3
2
3

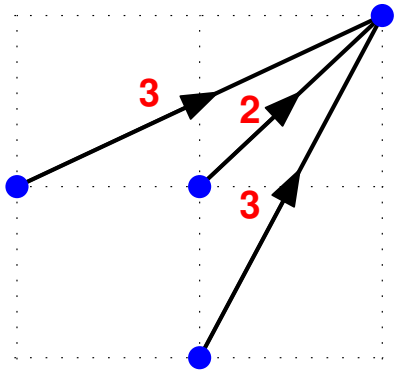
Inf

Inf

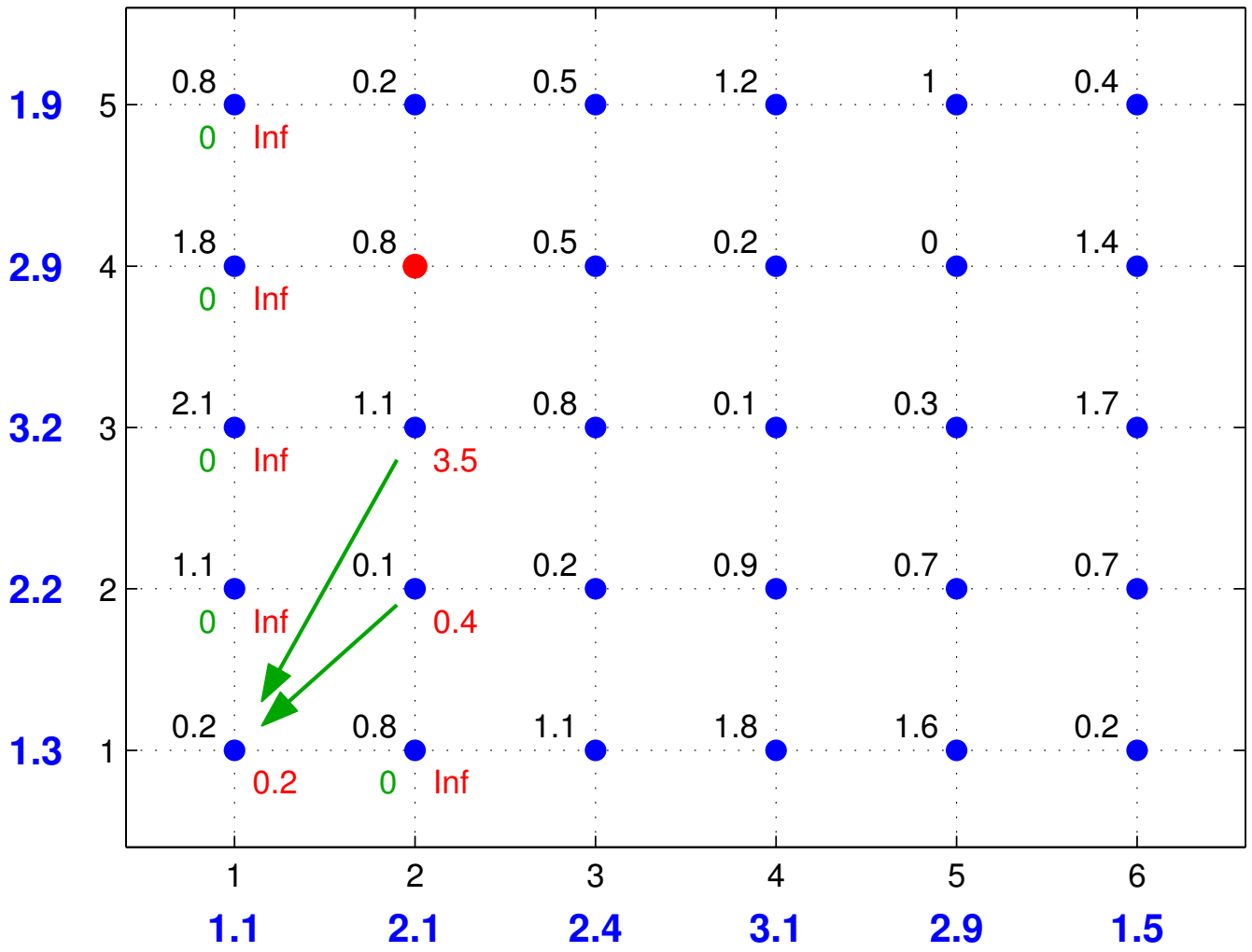
3.5

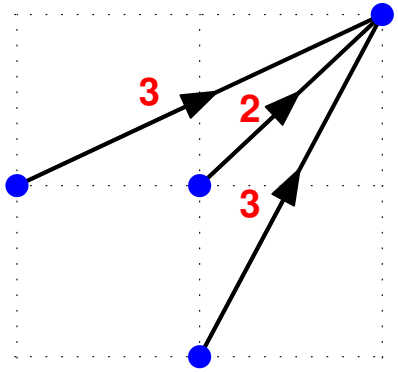
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

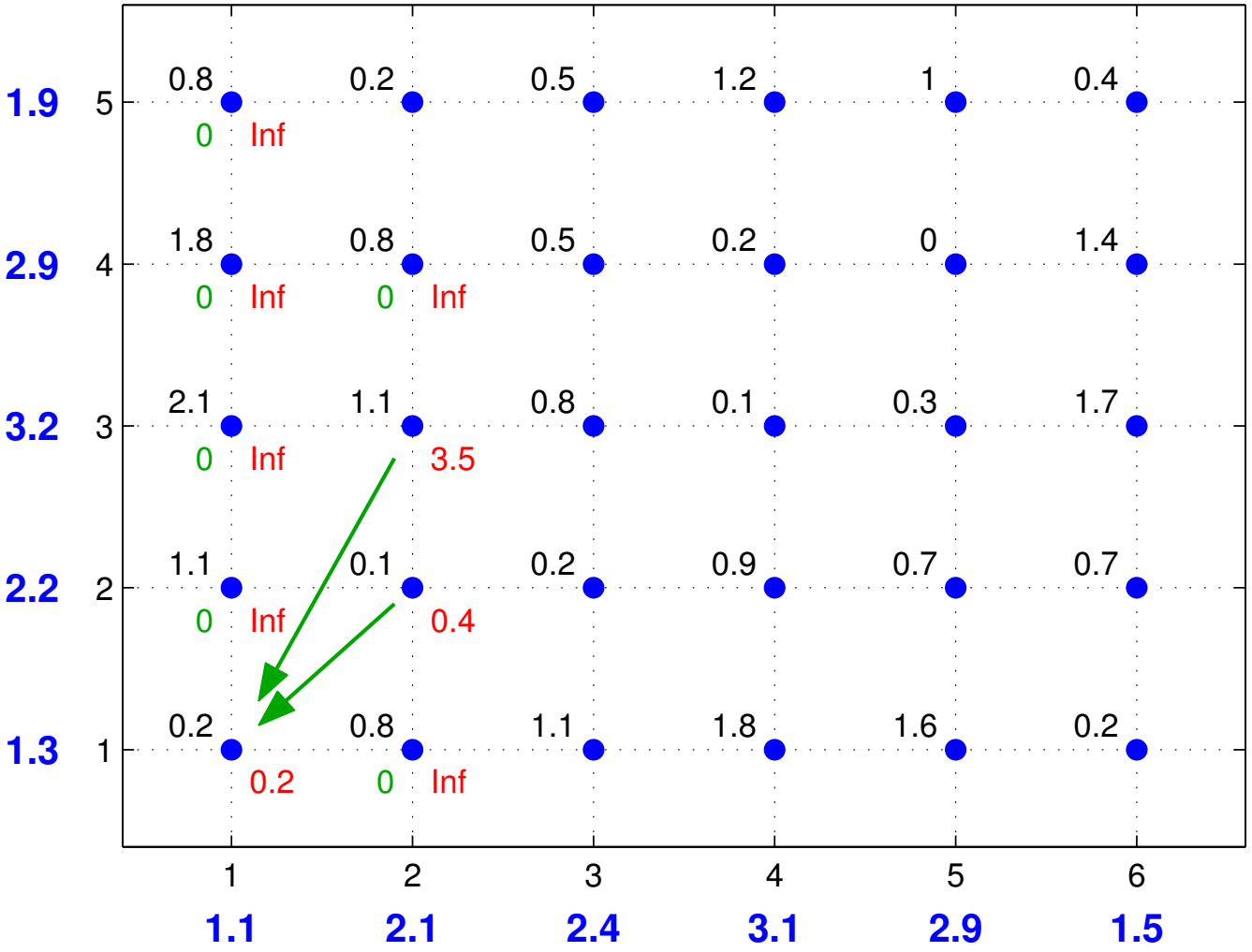

3
2
3

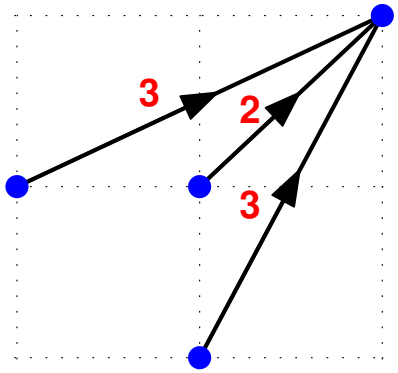
Inf

Inf

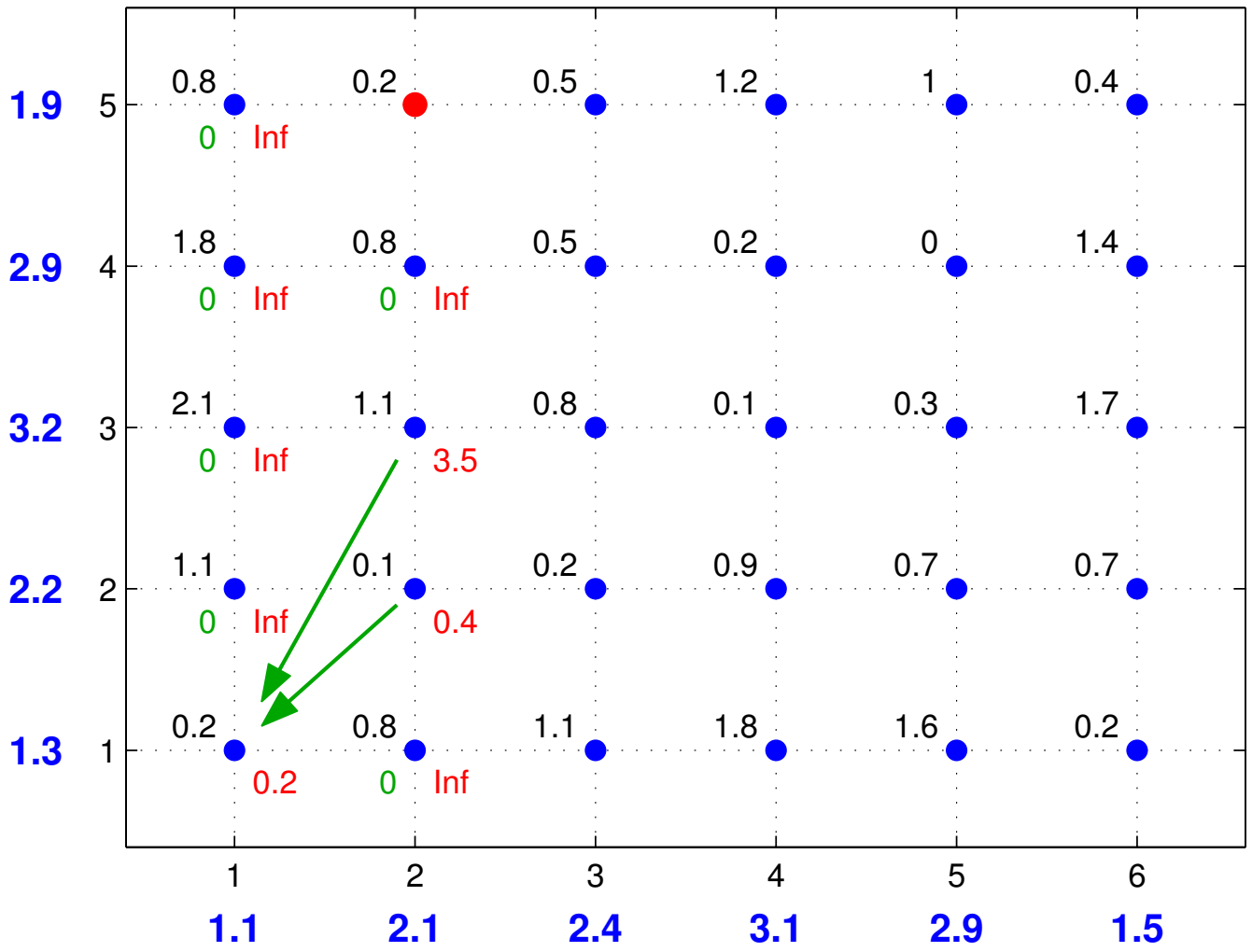
Inf

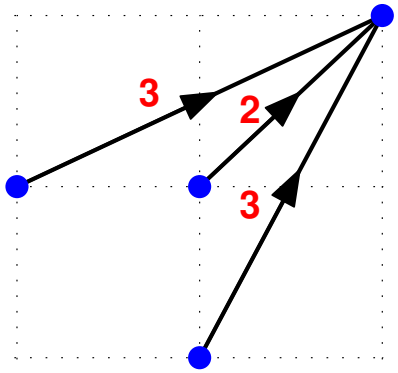
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf 0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf 0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

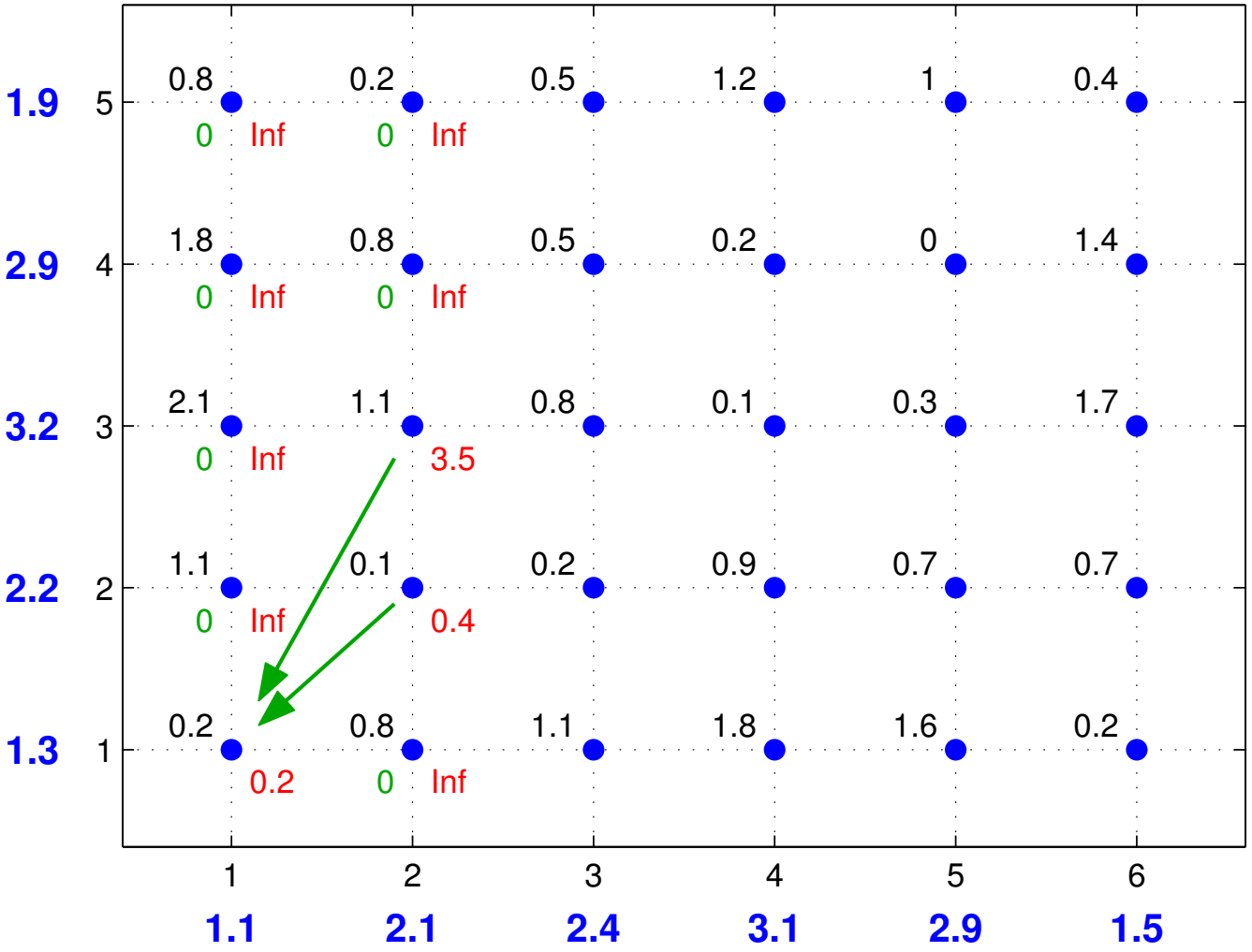

3
2
3

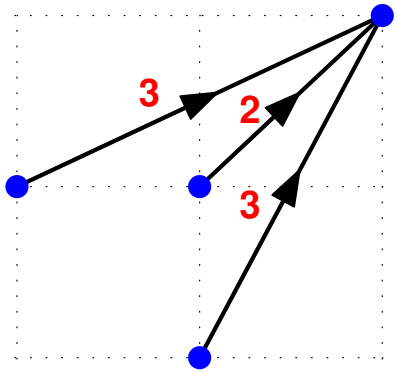
Inf

Inf

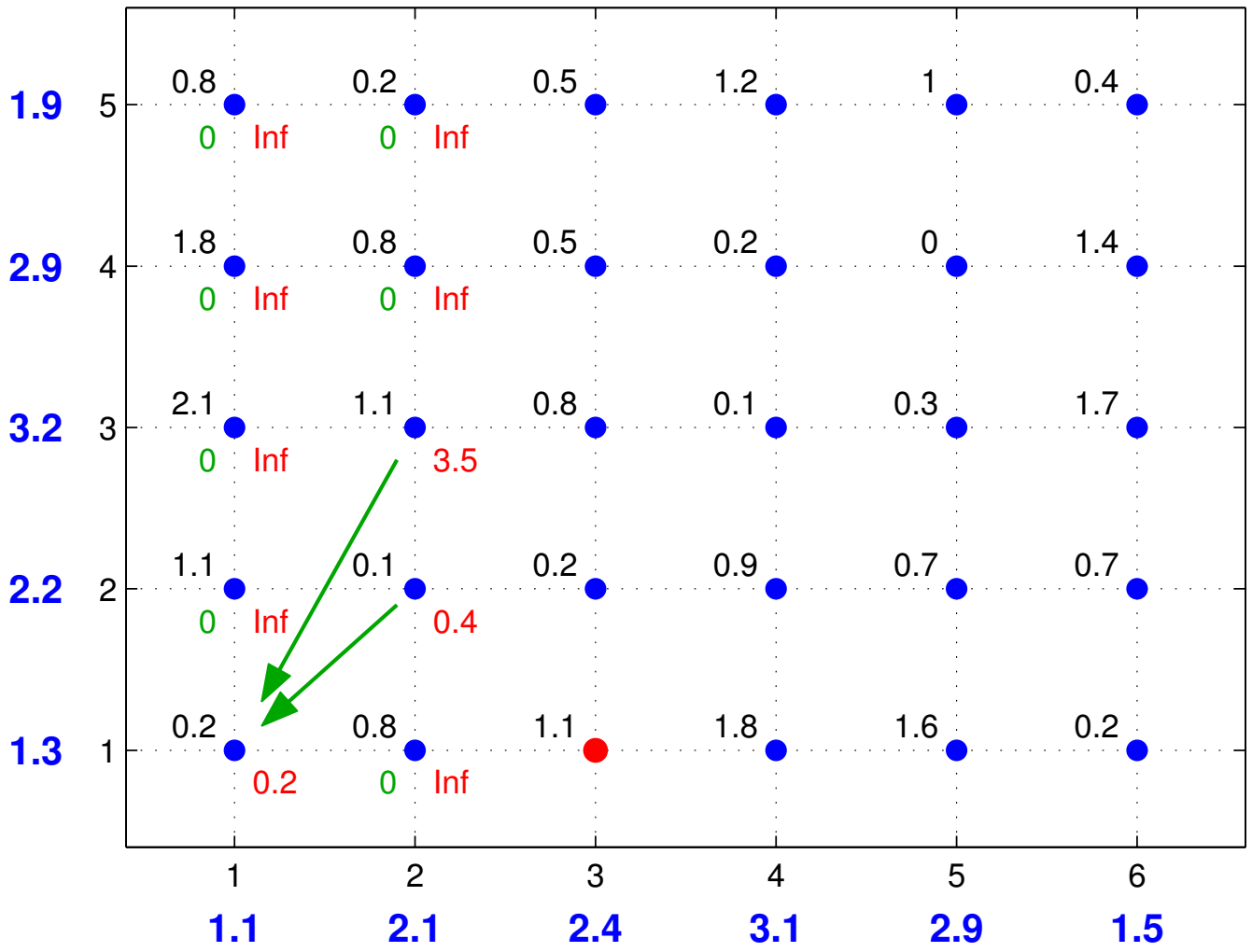
Inf

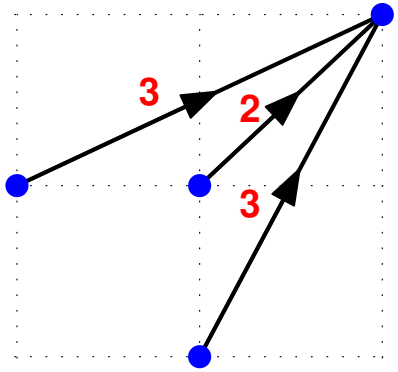
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf 0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf 0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
0 Inf 0 Inf
0 Inf 0 Inf
0 Inf 3.5
0 Inf 0.4
0.2 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

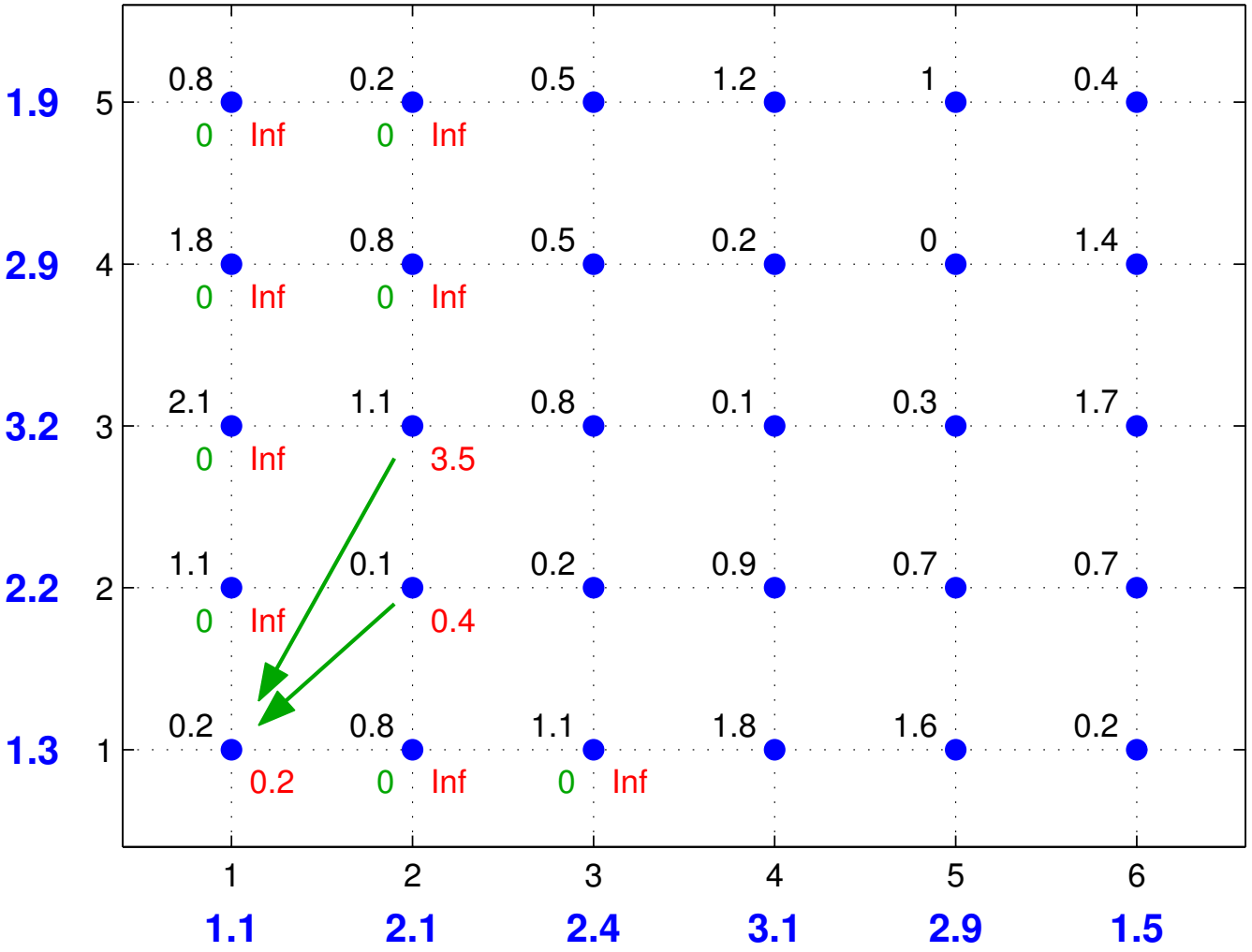

3
2
3

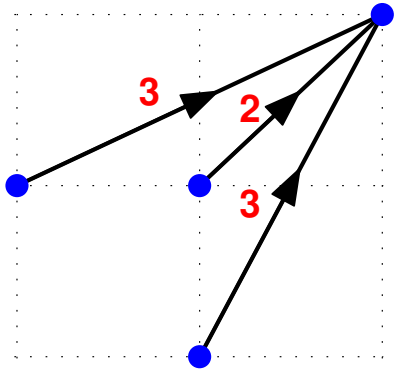
Inf

Inf

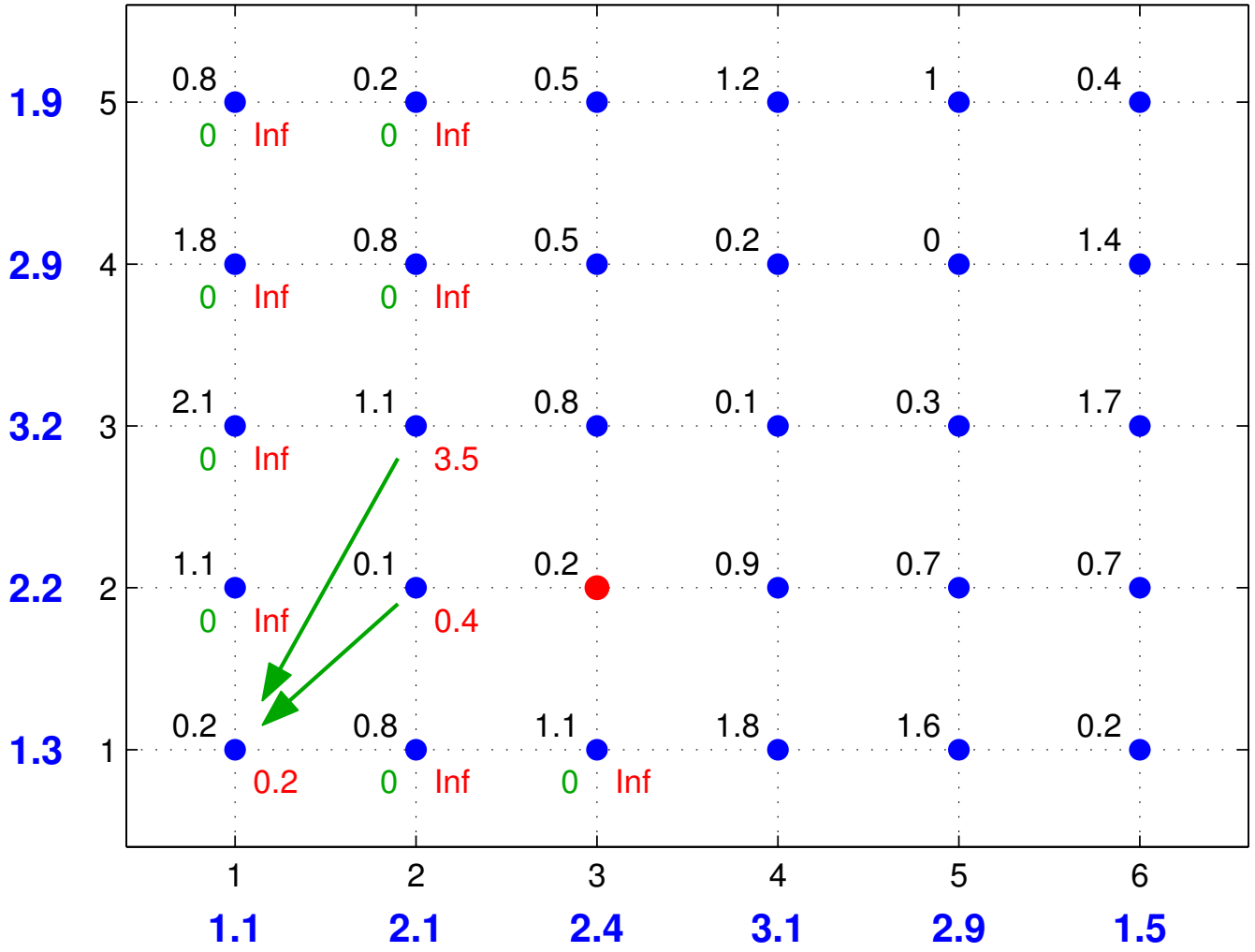
Inf

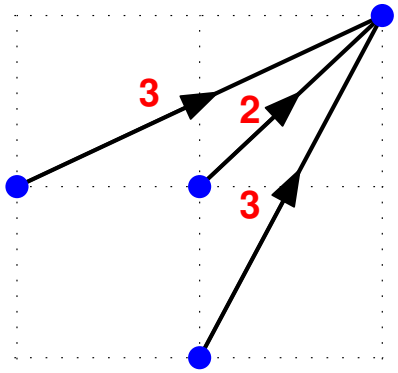
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf 0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf 0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
0 Inf 0 Inf
0 Inf 0 Inf
0 Inf 3.5
0 Inf 0.4
0.2 0 Inf 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

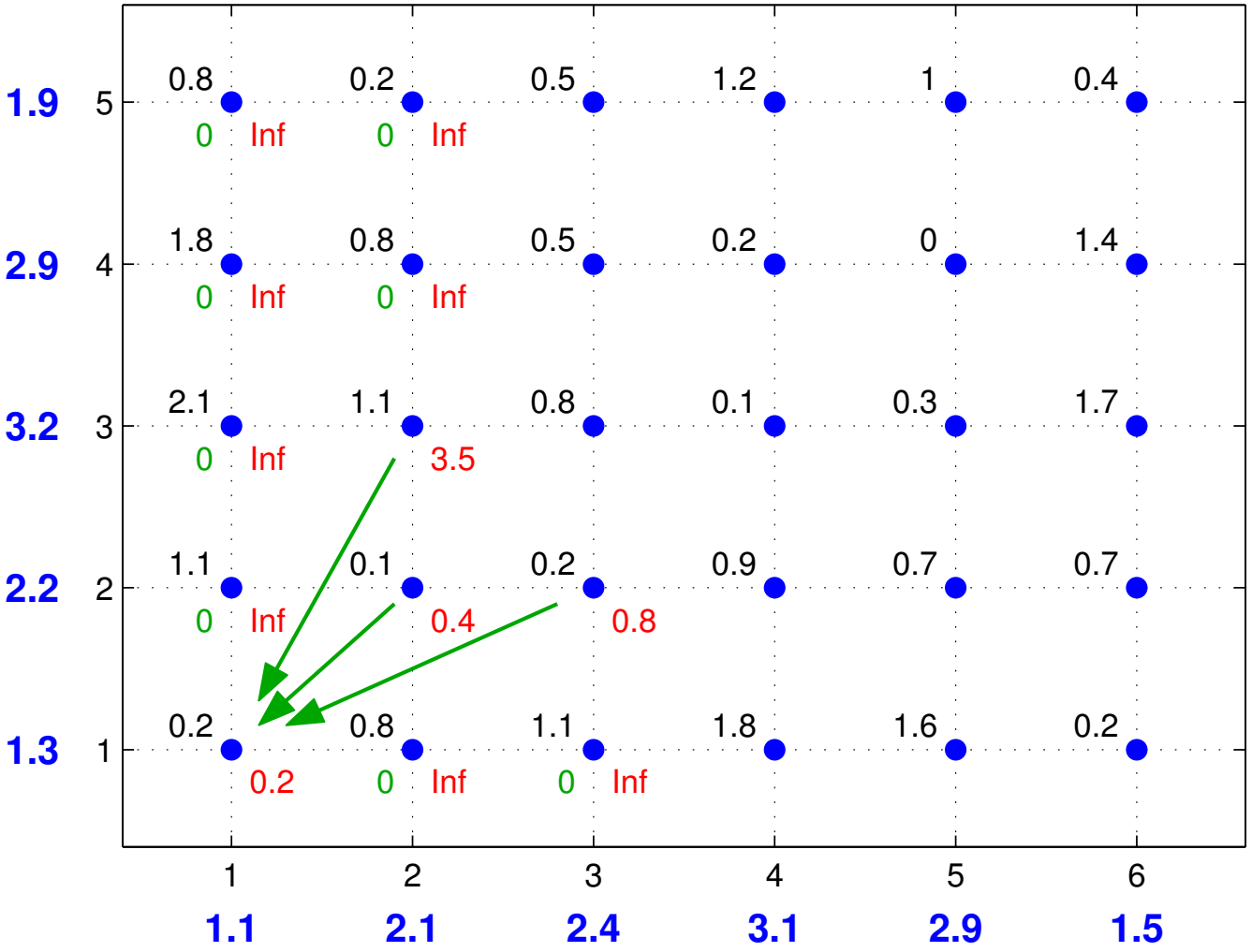

3
2
3

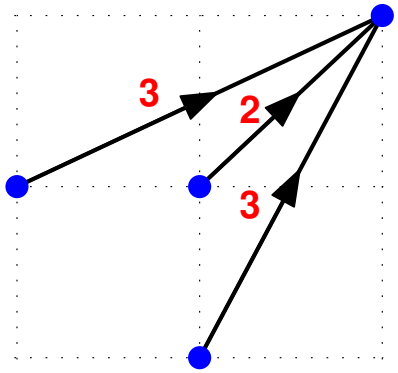
0.8

Inf

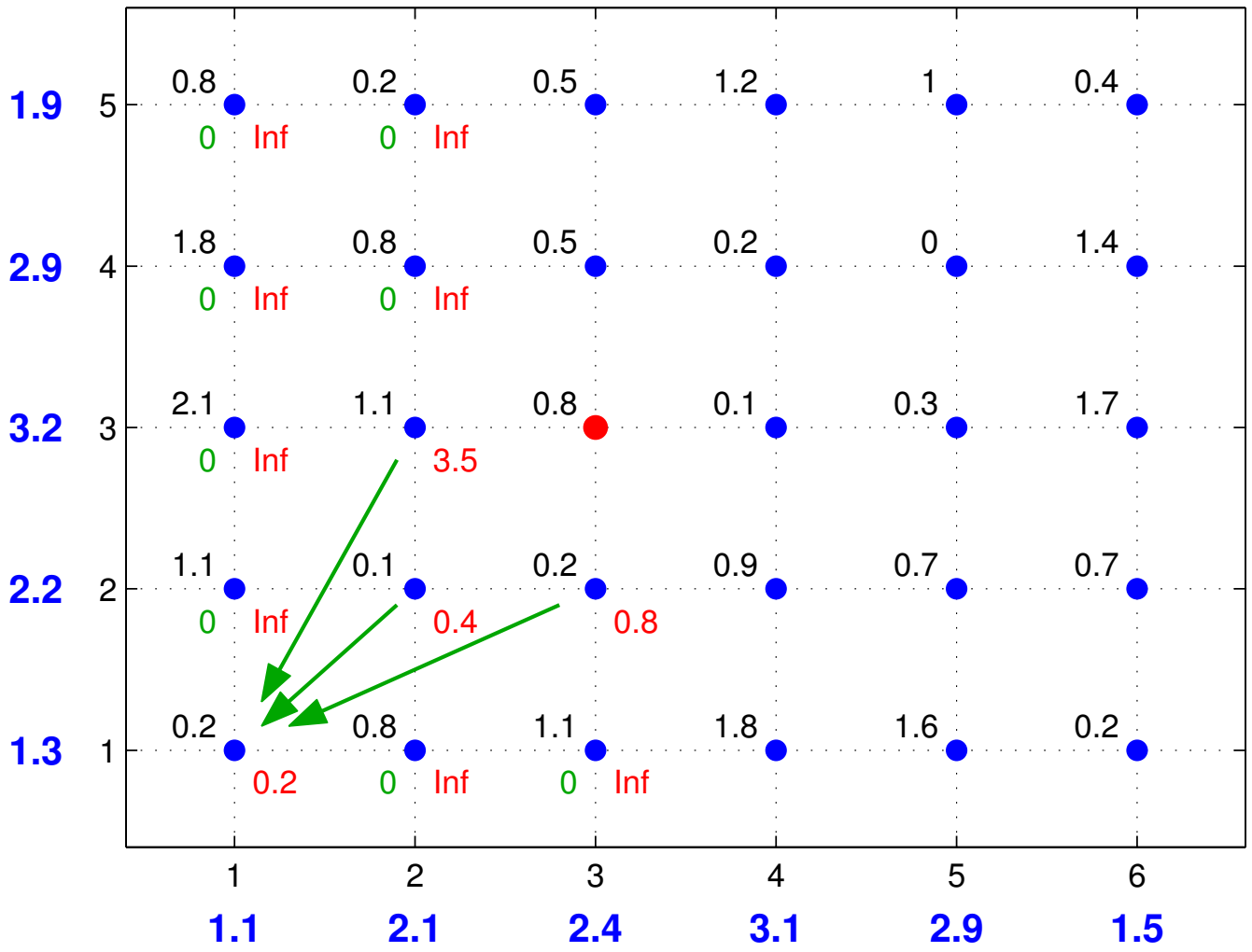
Inf

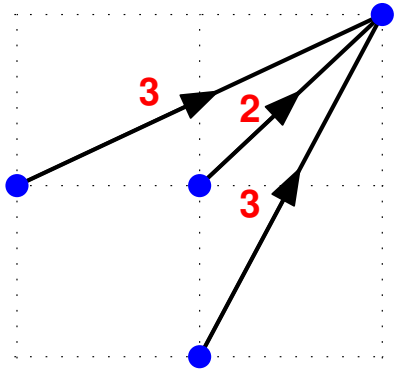
>>>


1.9
2.9
3.2
2.2
1.3
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
0 Inf 0 Inf
0 Inf 0 Inf
0 Inf 3.5
0 Inf 0.4 0.8
0.2 0 Inf 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
0 Inf 0 Inf
0 Inf 0 Inf
0 Inf 3.5
0 Inf 0.4 0.8
0.2 0 Inf 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

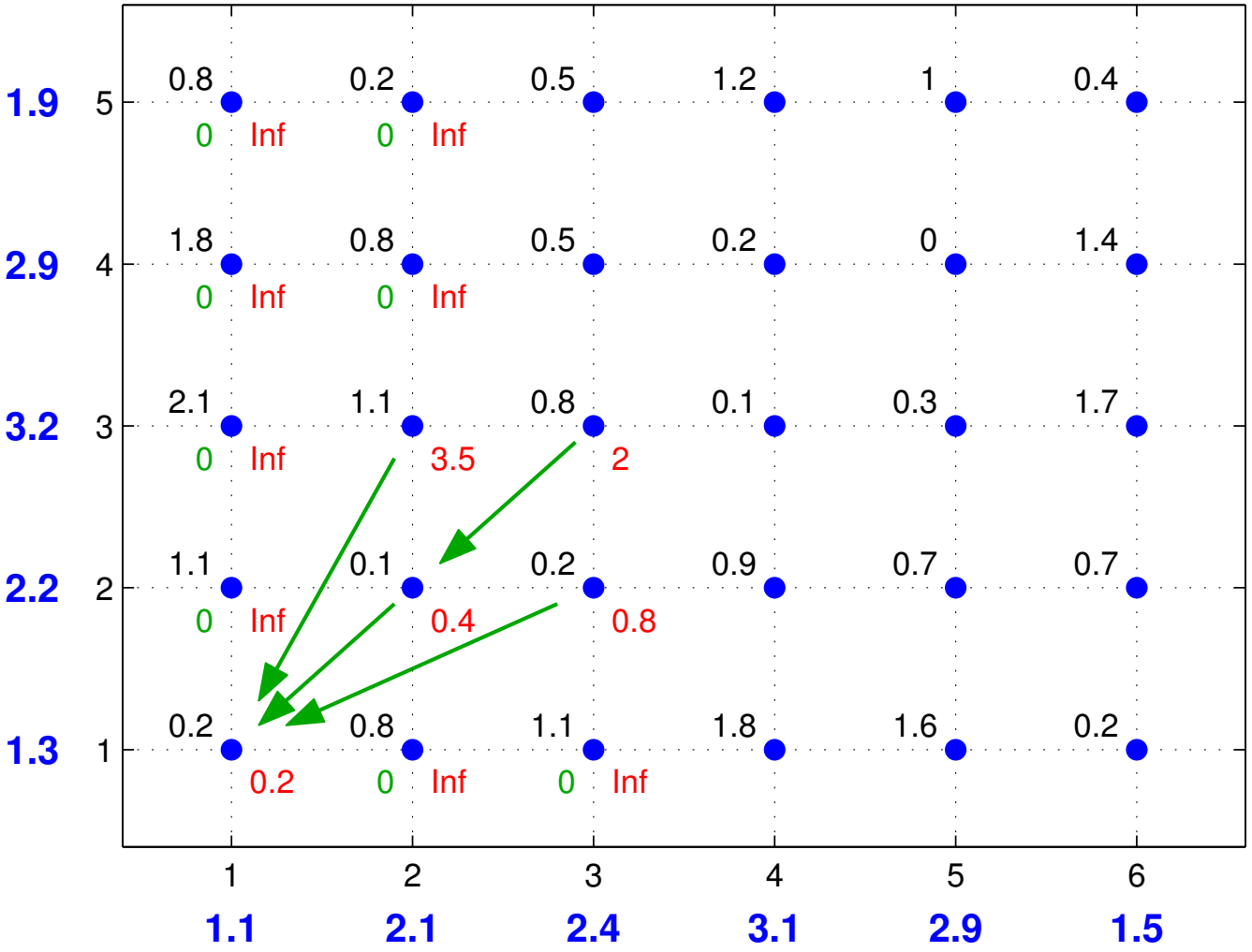

3
2
3

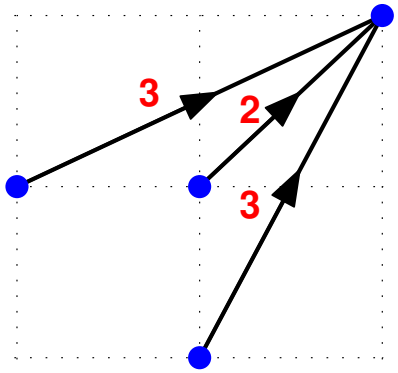
Inf

2

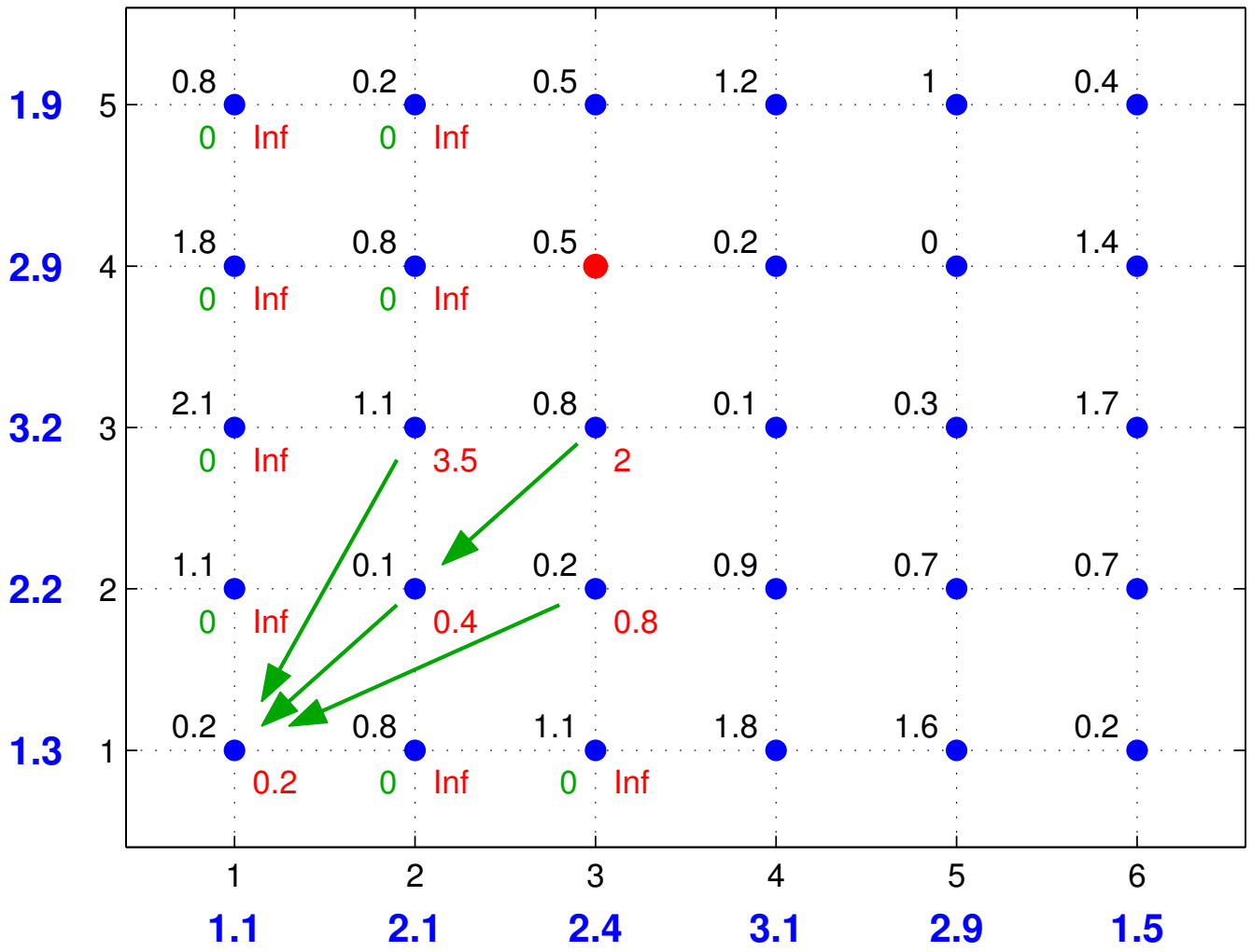
Inf

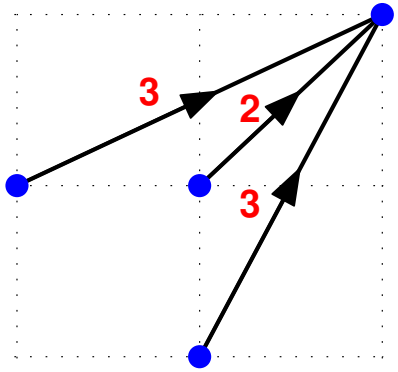
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf 0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf 0 Inf
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5 2
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4 0.8
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>


1.9
2.9
3.2
2.2
1.3
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
0 Inf 0 Inf
0 Inf 0 Inf
0 Inf 3.5 2
0 Inf 0.4 0.8
0.2 0 Inf 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

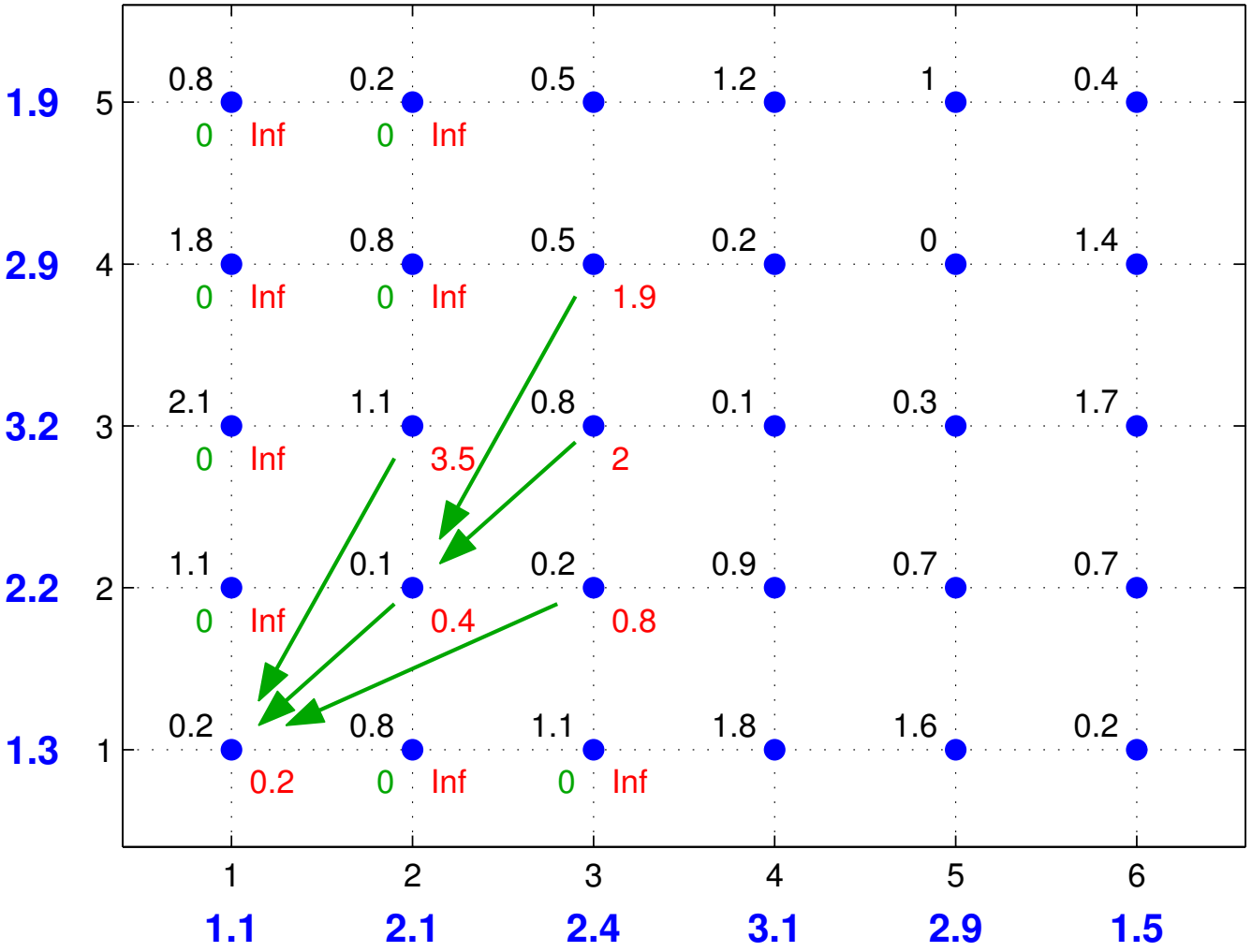

3
2
3

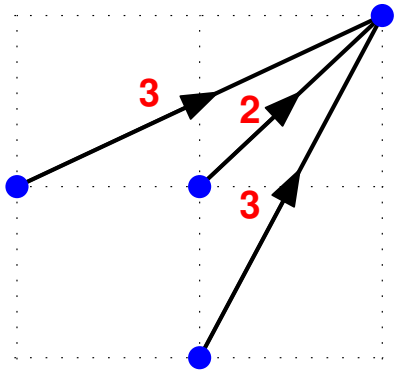
Inf

4.5

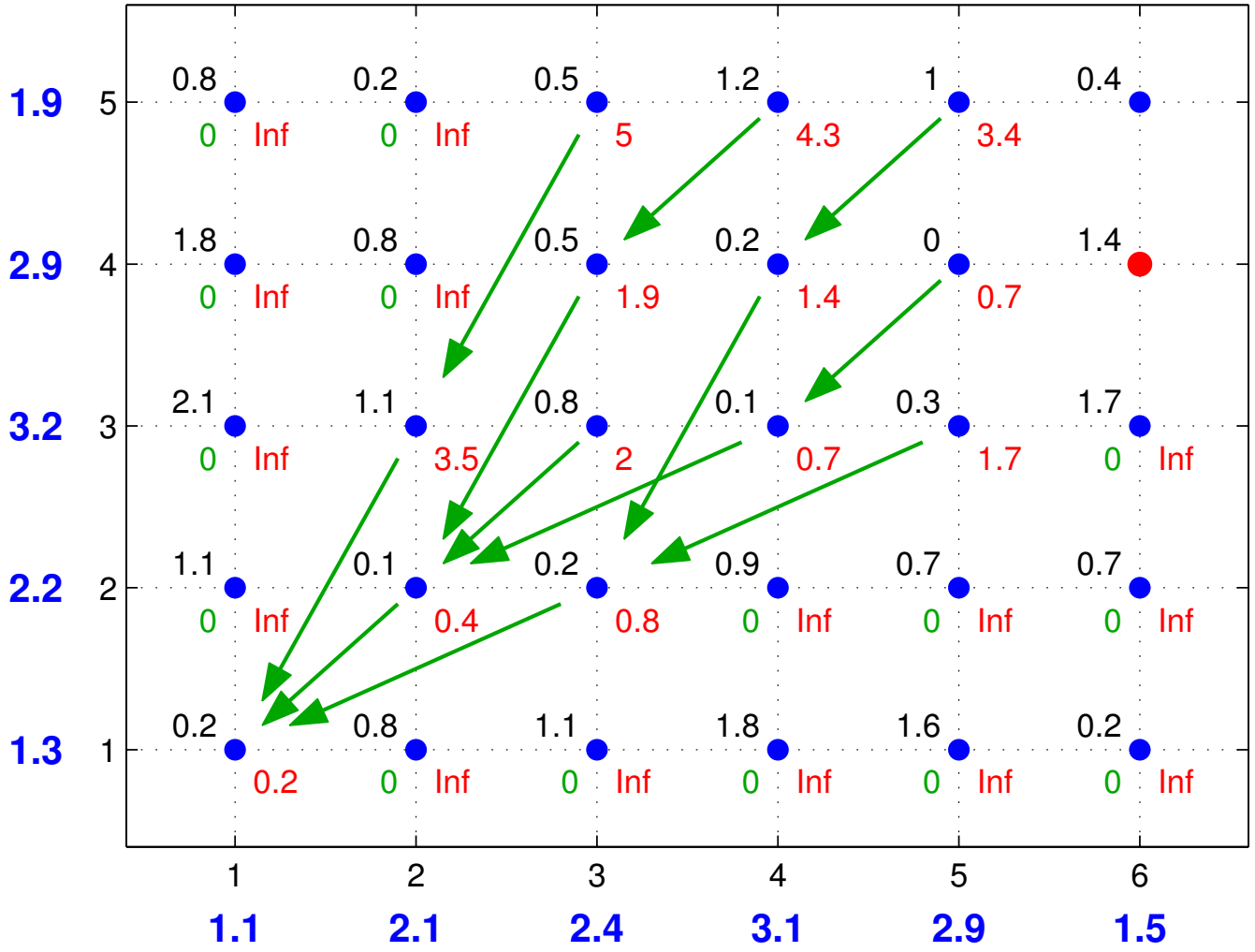
1.9

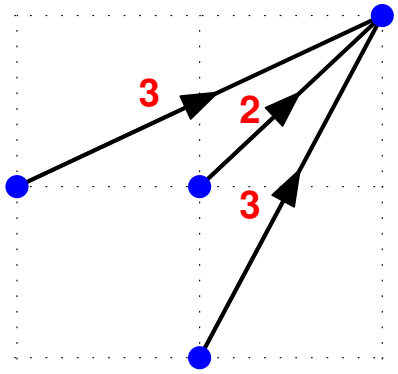
>>>


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf 0 Inf
1.8 0.8 0.5 0.2 0 1.4
0 Inf 0 Inf 1.9
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5 2
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4 0.8
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

>>>

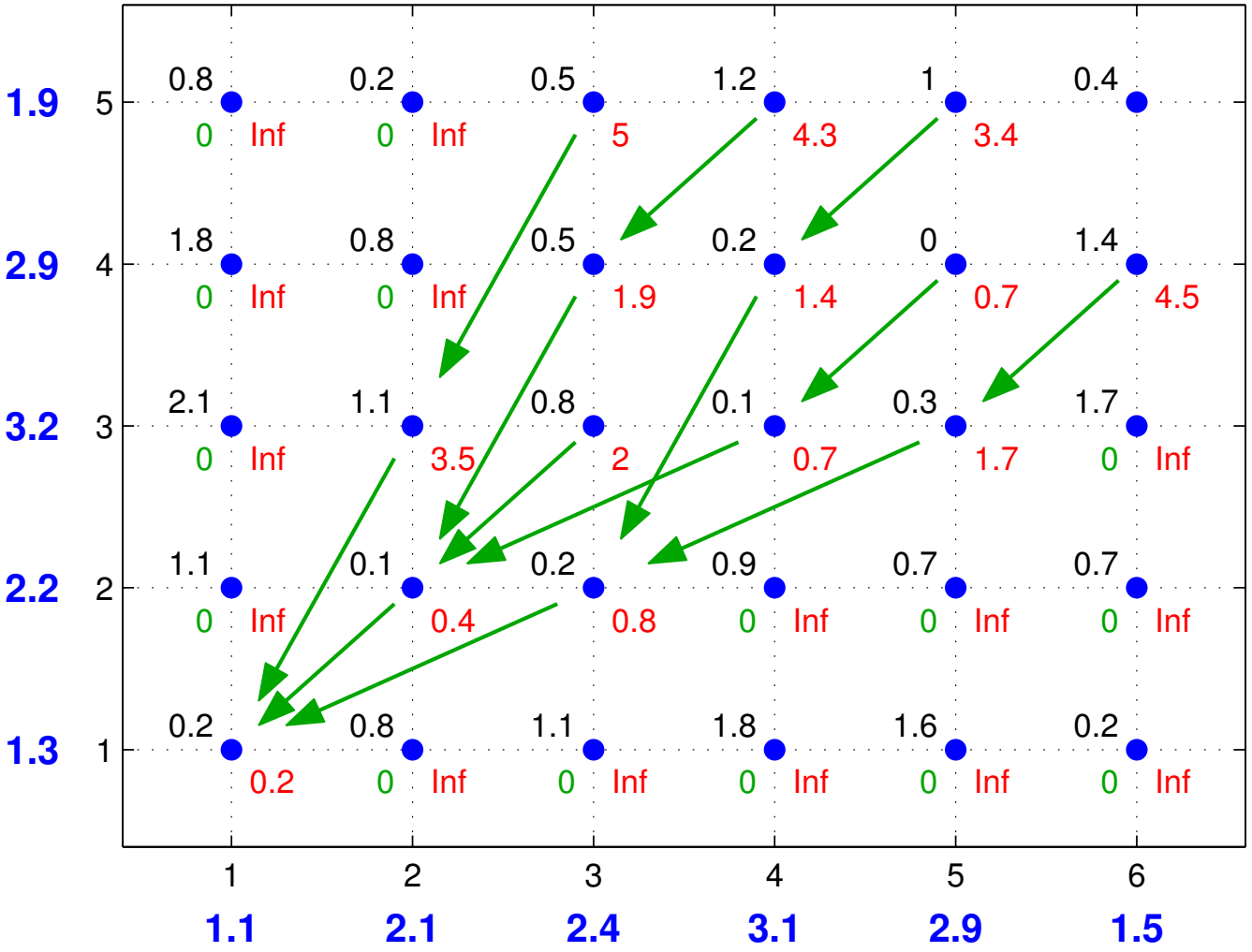

1.9
2.9
3.2
2.2
1.3
1.1
2.1
2.4
3.1
2.9
1.5
0.8 0.2 0.5 1.2 1 0.4
1.8 0.8 0.5 0.2 0 1.4
2.1 1.1 0.8 0.1 0.3 1.7
1.1 0.1 0.2 0.9 0.7 0.7
0.2 0.8 1.1 1.8 1.6 0.2
Inf
5
4.3
3.4
1.9
1.4
0.7
3.5
2
0.7
1.7
0.4
0.8
0.2

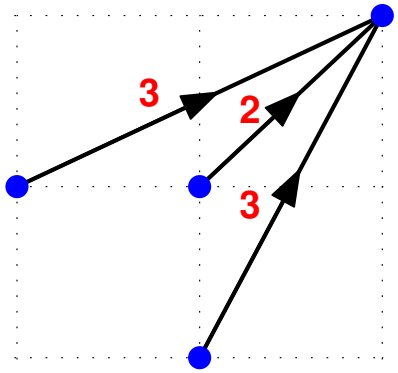

3
2
3

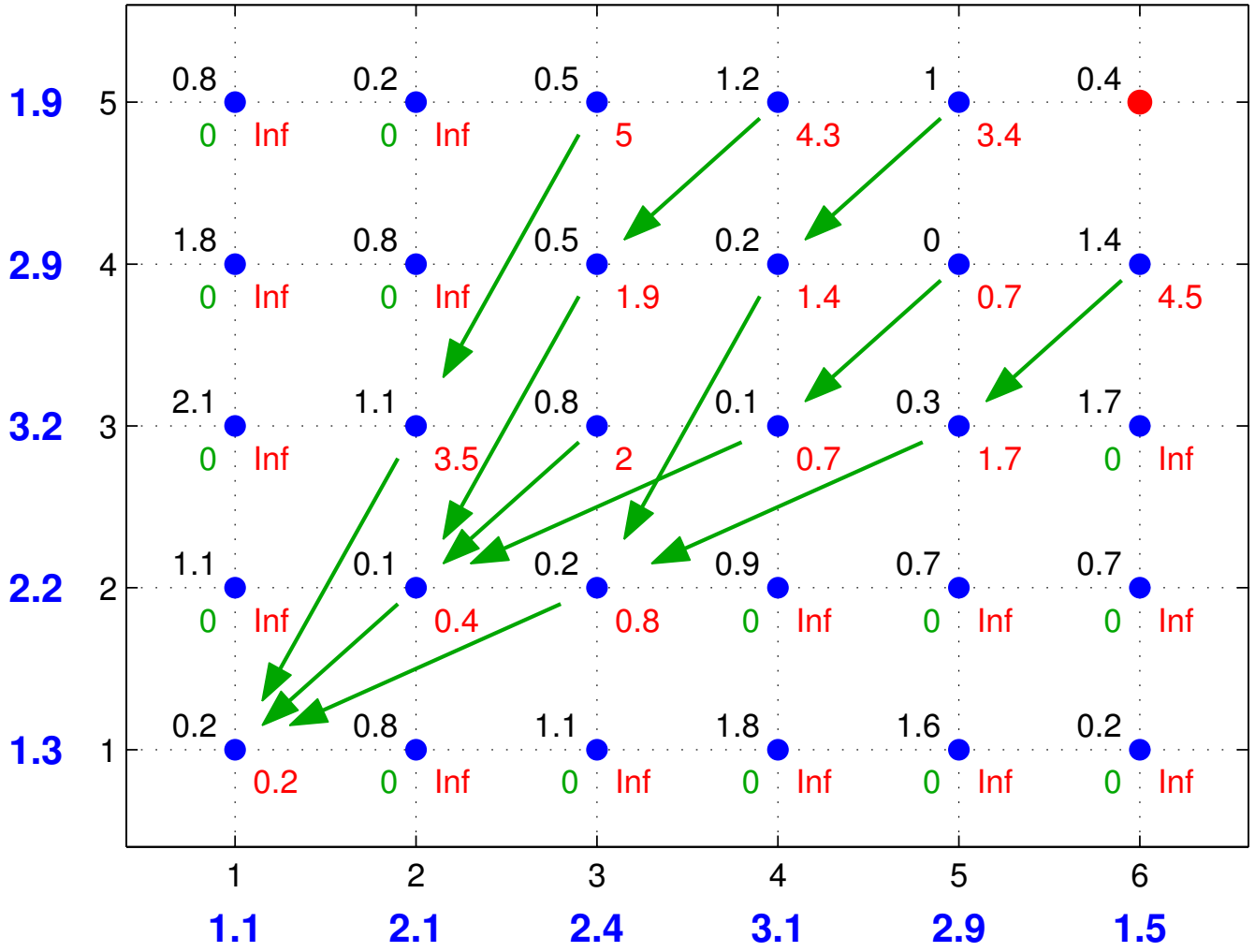
4.9

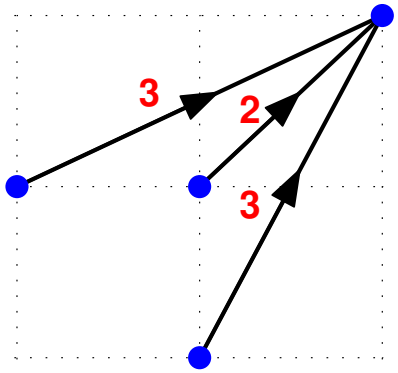
4.5

Inf


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf 0 Inf 5 4.3 3.4
1.8 0.8 0.5 0.2 0 1.4
0 Inf 0 Inf 1.9 1.4 0.7 4.5
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5 2 0.7 1.7 0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4 0.8 0 Inf 0 Inf 0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf 0 Inf 0 Inf 0 Inf 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

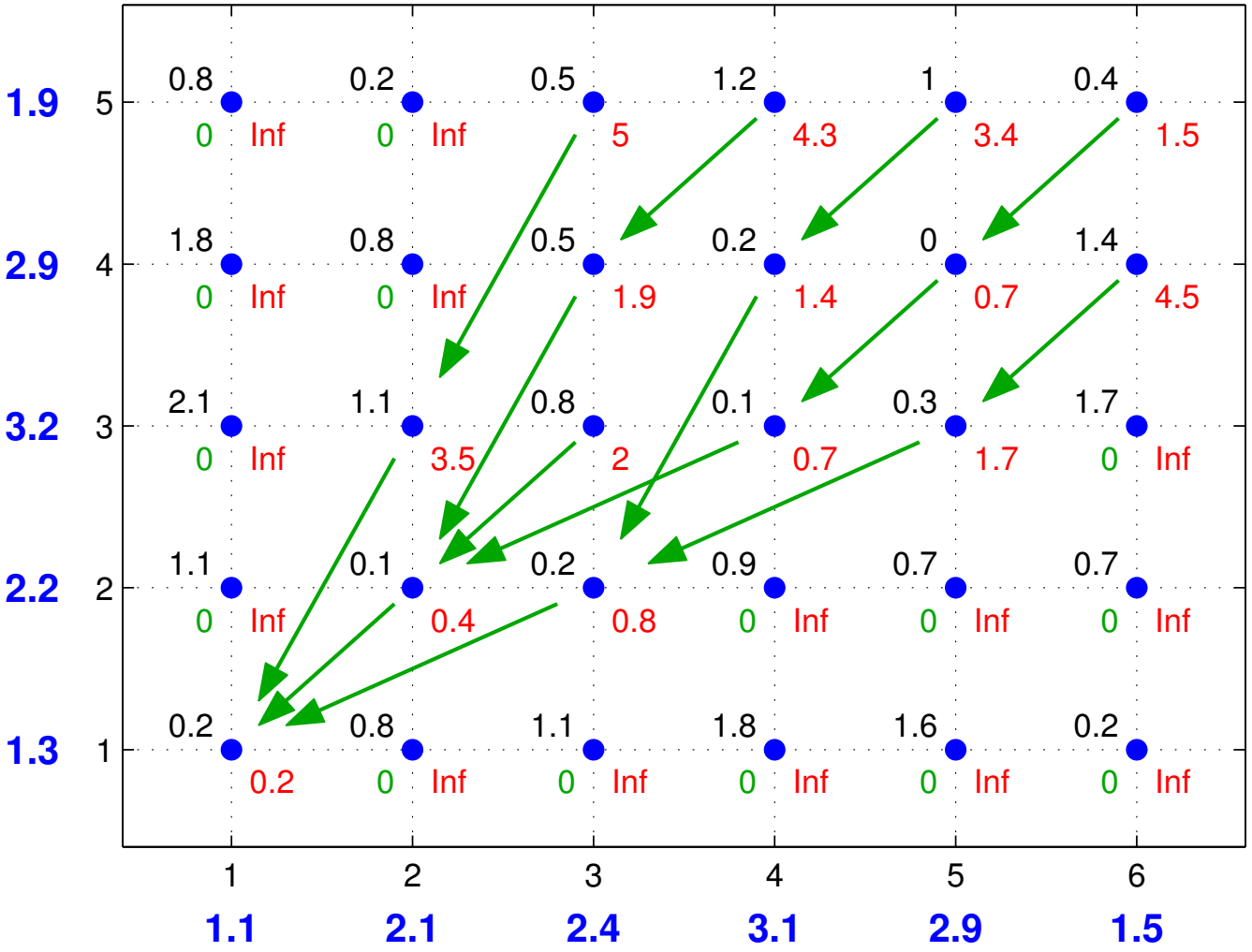

1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf 0 Inf 5 4.3 3.4
1.8 0.8 0.5 0.2 0 1.4
0 Inf 0 Inf 1.9 1.4 0.7 4.5
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5 2 0.7 1.7 0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4 0.8 0 Inf 0 Inf 0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf 0 Inf 0 Inf 0 Inf 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

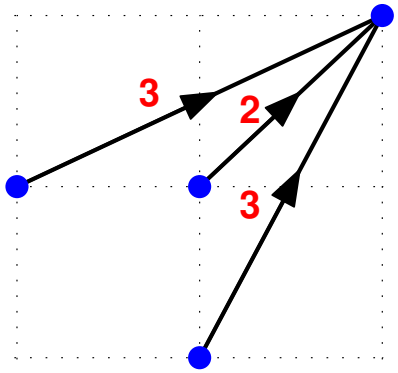

3
2
3

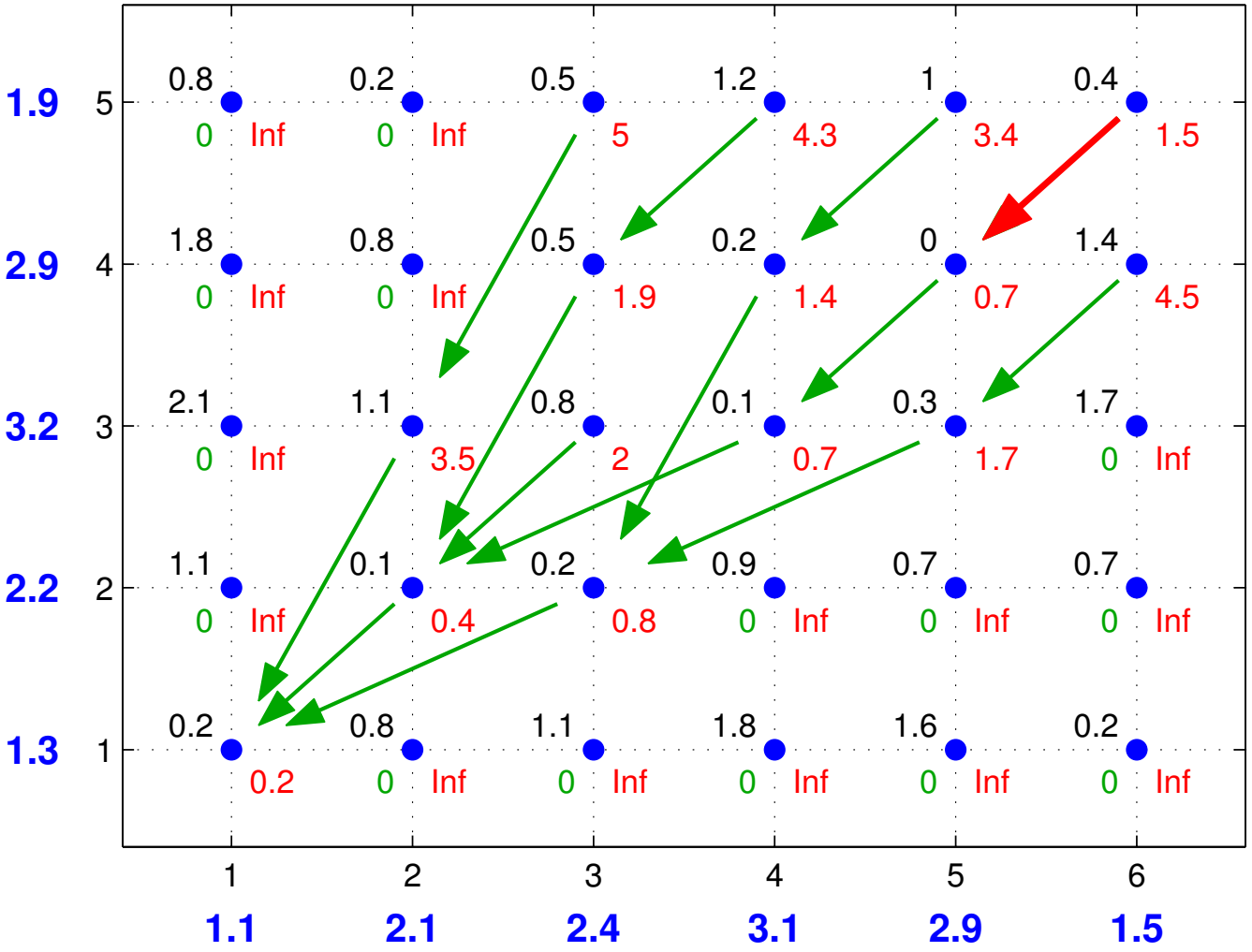
2.6

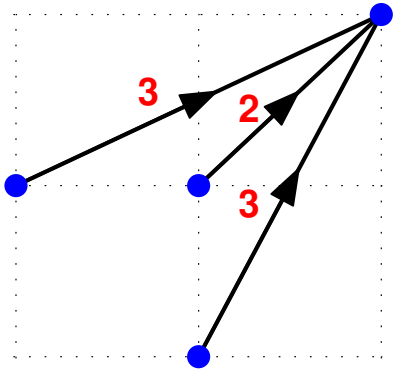
1.5

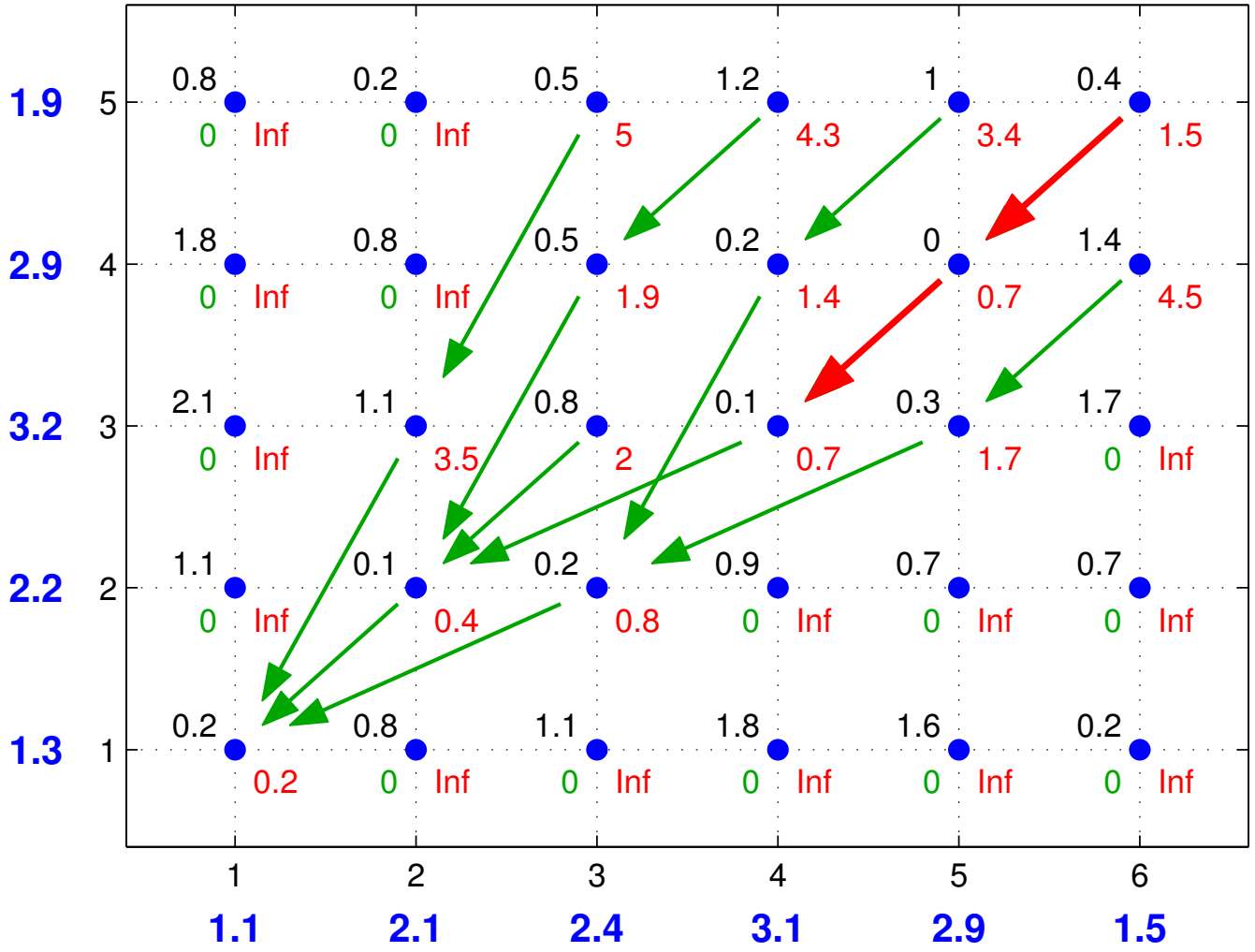
2.9

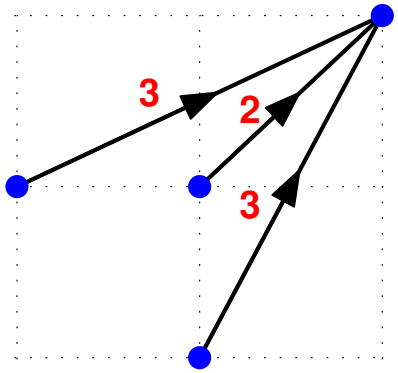

1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf 0 Inf 5 4.3 3.4 1.5
1.8 0.8 0.5 0.2 0 1.4
0 Inf 0 Inf 1.9 1.4 0.7 4.5
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5 2 0.7 1.7 0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4 0.8 0 Inf 0 Inf 0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf 0 Inf 0 Inf 0 Inf 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5

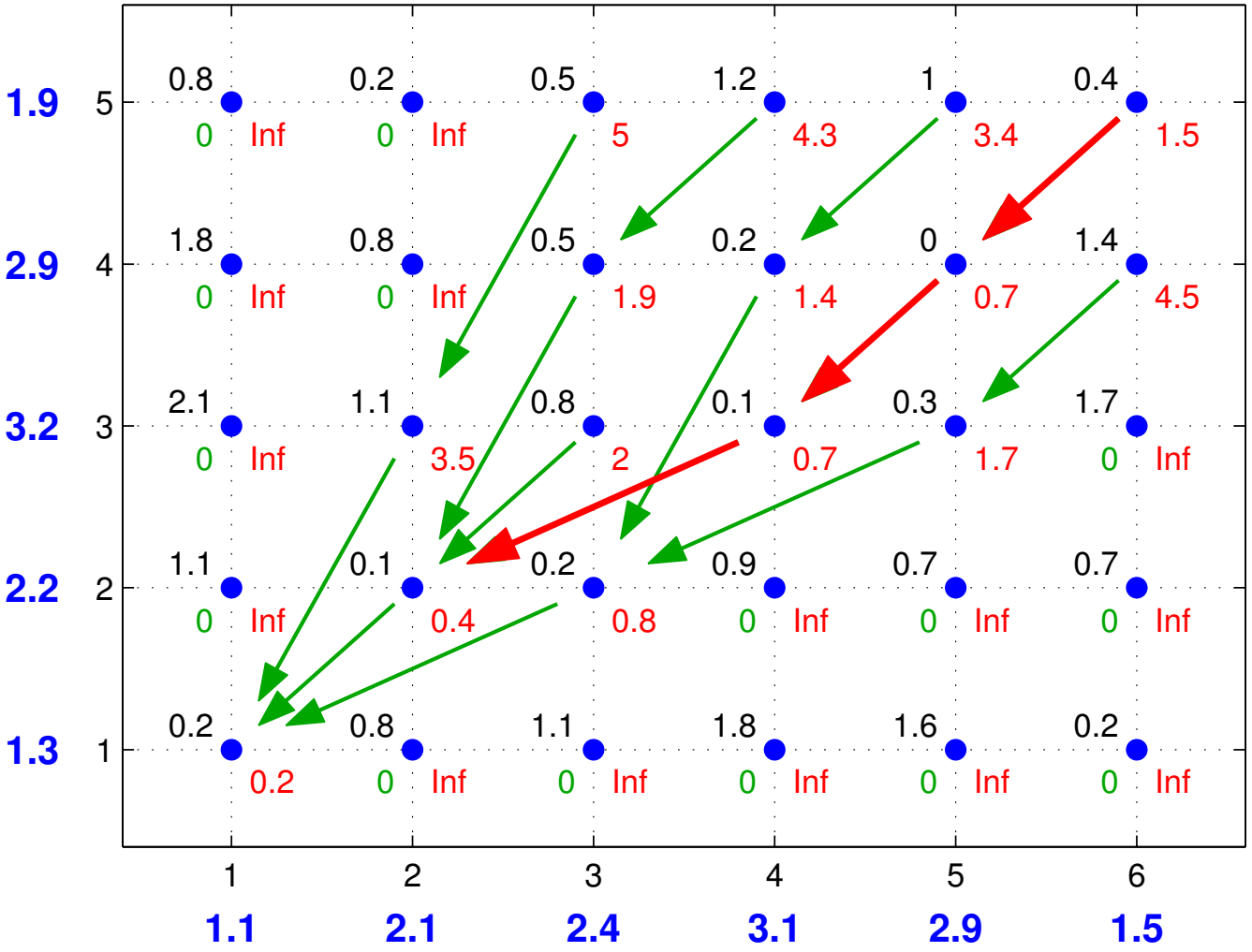

3
2
3

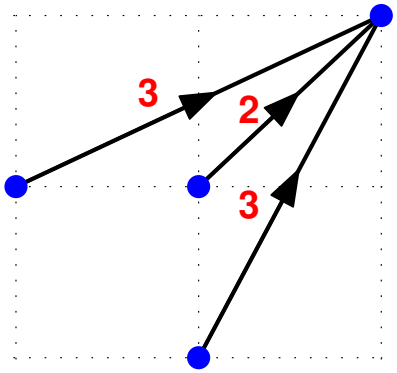

1.9
2.9
3.2
2.2
1.3
1.1
2.1
2.4
3.1
2.9
1.5

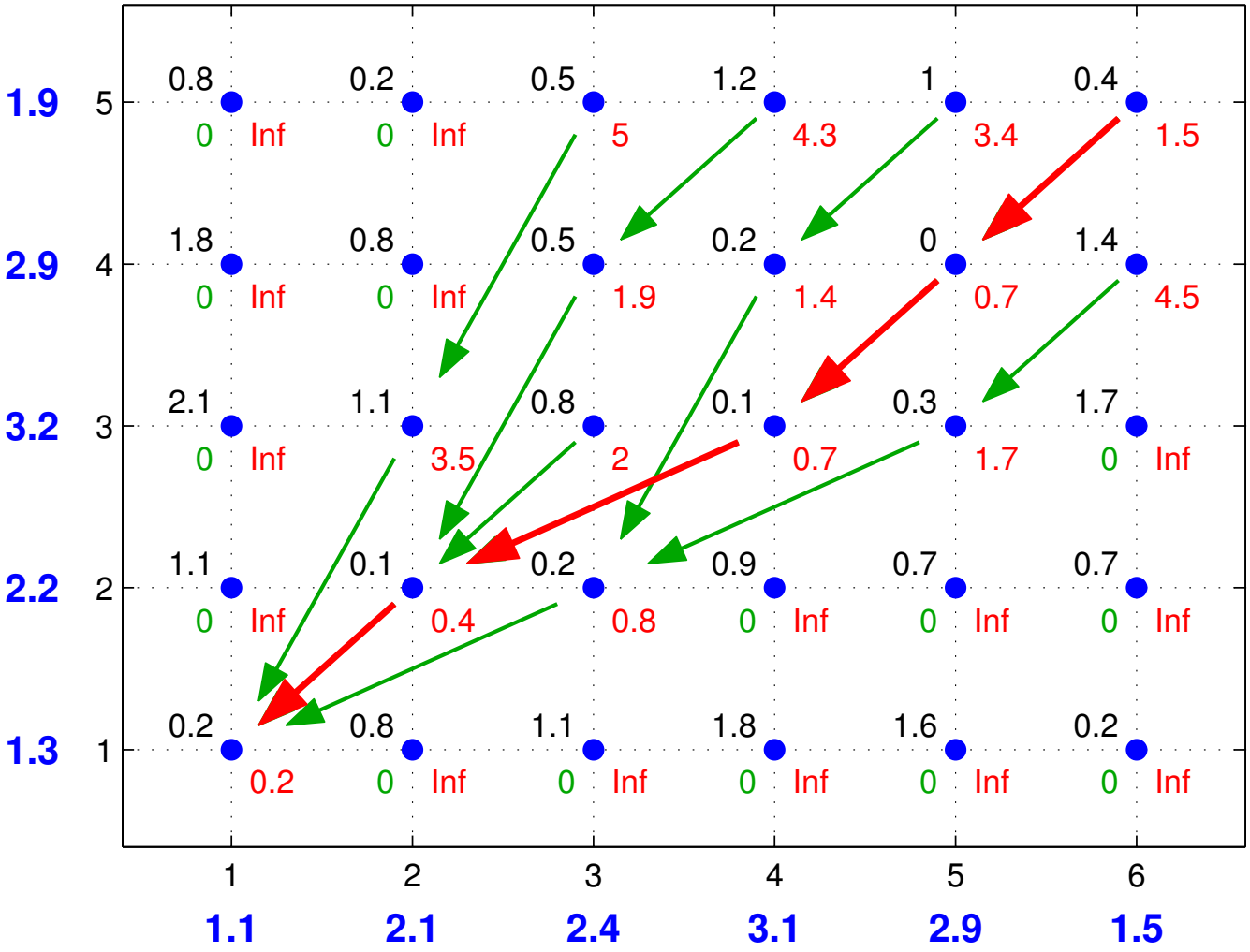

3
2
3

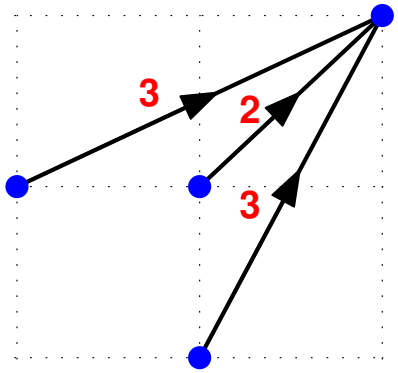

1.9
2.9
3.2
2.2
1.3
1.1
2.1
2.4
3.1
2.9
1.5


3
2
3


1.9
2.9
3.2
2.2
1.3
1.1
2.1
2.4
3.1
2.9
1.5


3
2
3


1.9
2.9
3.2
2.2
1.3
5
4
3
2
1
0.8 0.2 0.5 1.2 1 0.4
0 Inf 0 Inf 5 4.3 3.4 1.5
1.8 0.8 0.5 0.2 0 1.4
0 Inf 0 Inf 1.9 1.4 0.7 4.5
2.1 1.1 0.8 0.1 0.3 1.7
0 Inf 3.5 2 0.7 1.7 0 Inf
1.1 0.1 0.2 0.9 0.7 0.7
0 Inf 0.4 0.8 0 Inf 0 Inf 0 Inf
0.2 0.8 1.1 1.8 1.6 0.2
0.2 0 Inf 0 Inf 0 Inf 0 Inf 0 Inf
1 2 3 4 5 6
1.1 2.1 2.4 3.1 2.9 1.5


3
2
3

<<<

Merkmale für die Sprachmuster s_x und s_y

Kurzzeitzeitanalyse $\longrightarrow$ Merkmalssequenz

Signal s_x: $\mathbf{X} = \mathbf{x}_1\, \mathbf{x}_2 \ldots \mathbf{x}_{T_x}$ mit $\mathbf{x}_i = \begin{pmatrix} \bar{c}_{xi}(0) \\ \bar{c}_{xi}(1) \\ \vdots \\ \bar{c}_{xi}(D) \end{pmatrix}$

Signal s_y: $\mathbf{Y} = \mathbf{y}_1\, \mathbf{y}_2 \ldots \mathbf{y}_{T_y}$ mit $\mathbf{y}_j = \begin{pmatrix} \bar{c}_{yj}(0) \\ \bar{c}_{yj}(1) \\ \vdots \\ \bar{c}_{yj}(D) \end{pmatrix}$

<<<

Distanzmass: Lokale Distanz

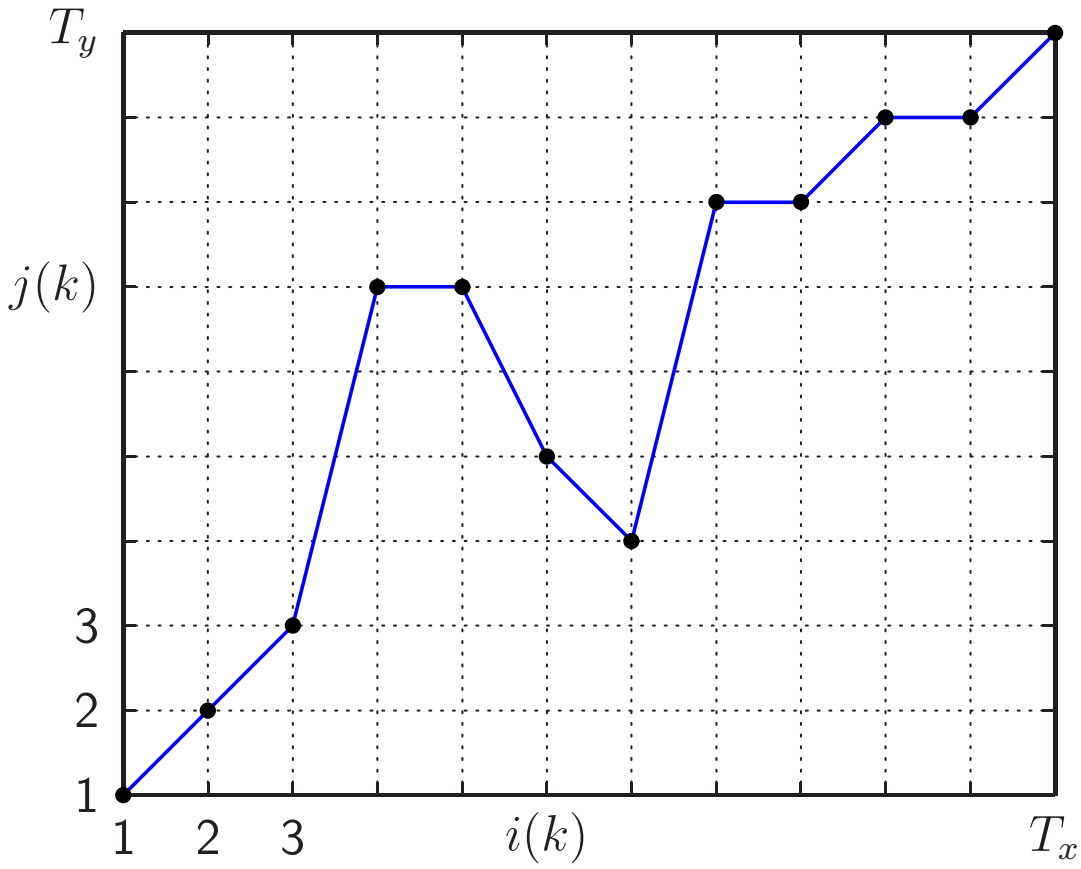
Unterschied zwischen den Merkmalsvektoren $\mathbf{x}_i$ und $\mathbf{y}_j$

Euklidische, mel-cepstrale Distanz:

$$d(\mathbf{x}_i, \mathbf{y}_j) = \sqrt{\sum_{m=1}^{D} \left[\bar{c}_{xi}(m) - \bar{c}_{yj}(m)\right]^2} \doteq d(i,j)$$

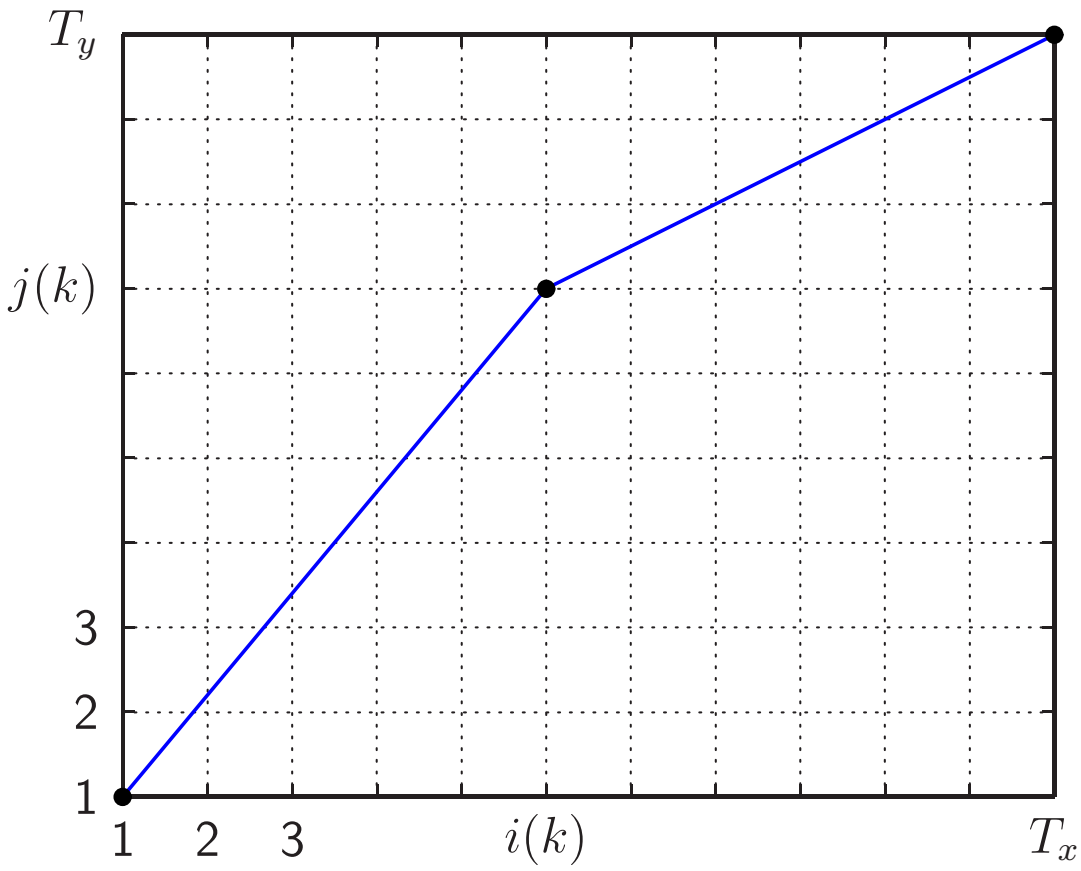
<<<

Monotonie



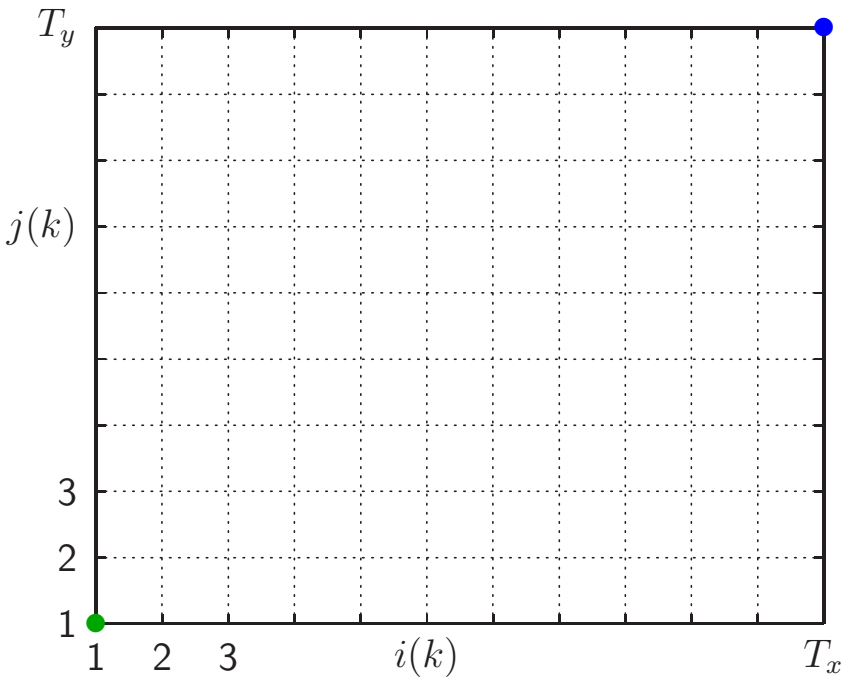
<<<

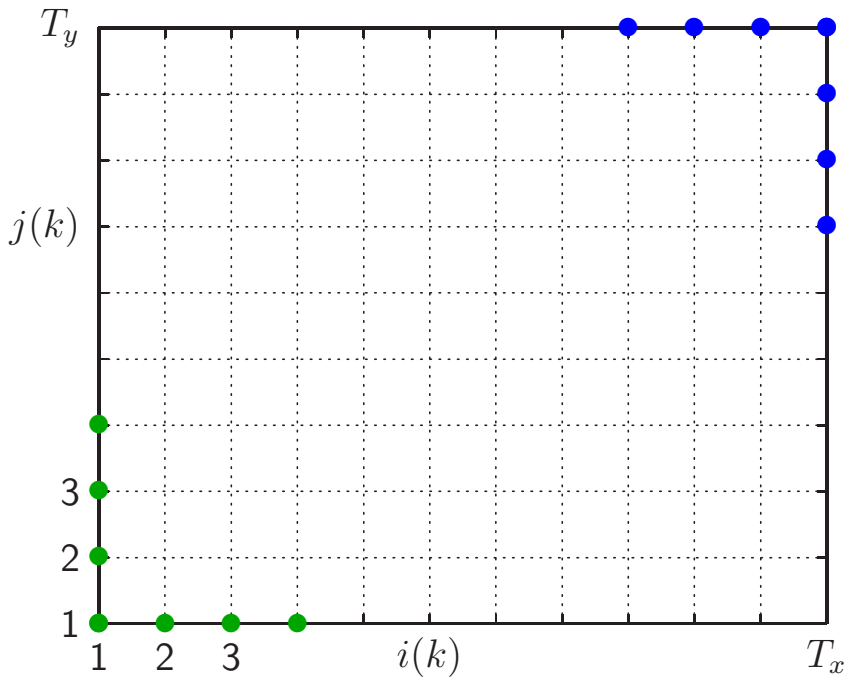
Lokale Kontinuität



<<<

Anfangs- und Endpunktbedingungen





<<<

Akkumulierte Distanz im Punkt (i, j)

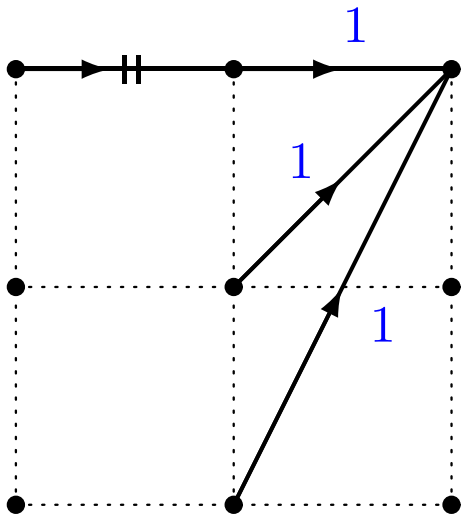
$$D_A(i,j) = \min \left\{ \begin{array}{l} D_A((i,j) - p(1)) + wp(1)\, d(i,j) \\ D_A((i,j) - p(2)) + wp(2)\, d(i,j) \\ \quad \vdots \\ D_A((i,j) - p(M)) + wp(M)\, d(i,j) \end{array} \right\}$$

<<<

Asymmetrische Zeitnormalisation

Muster $\mathbf{X}^{(1)}$ zeitlich an Muster $\mathbf{X}^{(r)}$ anpassen

$\longrightarrow$ spezielle Pfaderweiterungen nötig, z.B.:



$$D_A(i,j) = \min \left\{ \begin{array}{l} D_A(i-1,j)\, g(k) + d(i,j) \\ D_A(i-1,j-1) + d(i,j) \\ D_A(i-1,j-2) + d(i,j) \end{array} \right\},$$

wobei $g(k) = \left\{ \begin{array}{ll} \infty & \text{falls j(k-1)} = j(k-2) \\ 1 & \text{sonst} \end{array} \right.$

<<<