

ÁRVORES AVL

e
BUSCA EM
LARGURA

Alunos: Alexandre Nascimento, Caio Márcio,
Eloyse Fernanda e João Otogali

Árvores AVL

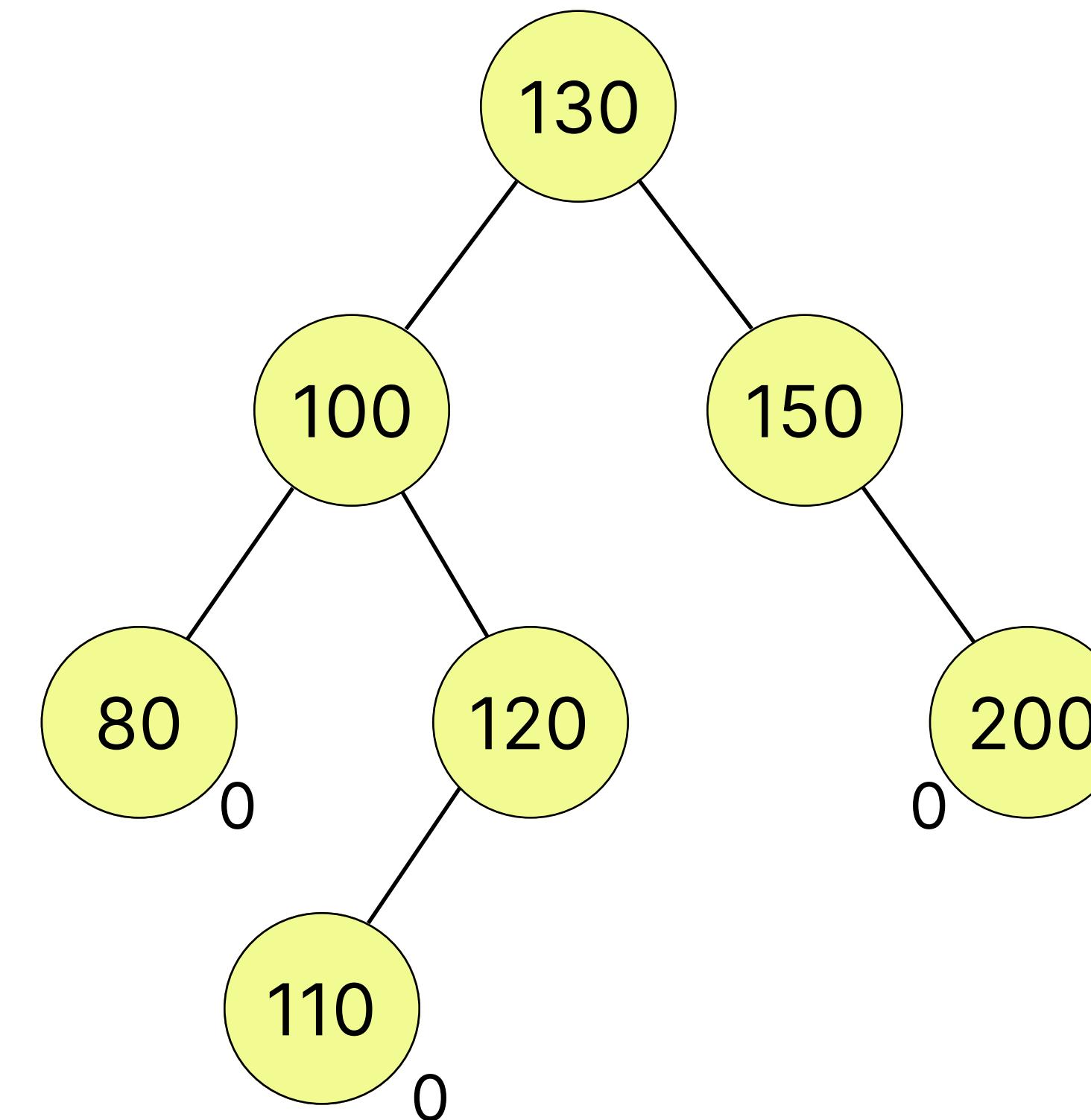
Contexto Histórico

- AVL (Adelson Velsky e Landis)
- Complexidade
- Pioneirismo

Fator de Balanceamento (FB)

Fator de balanceamento: diferença entre altura da subárvore direita e esquerda

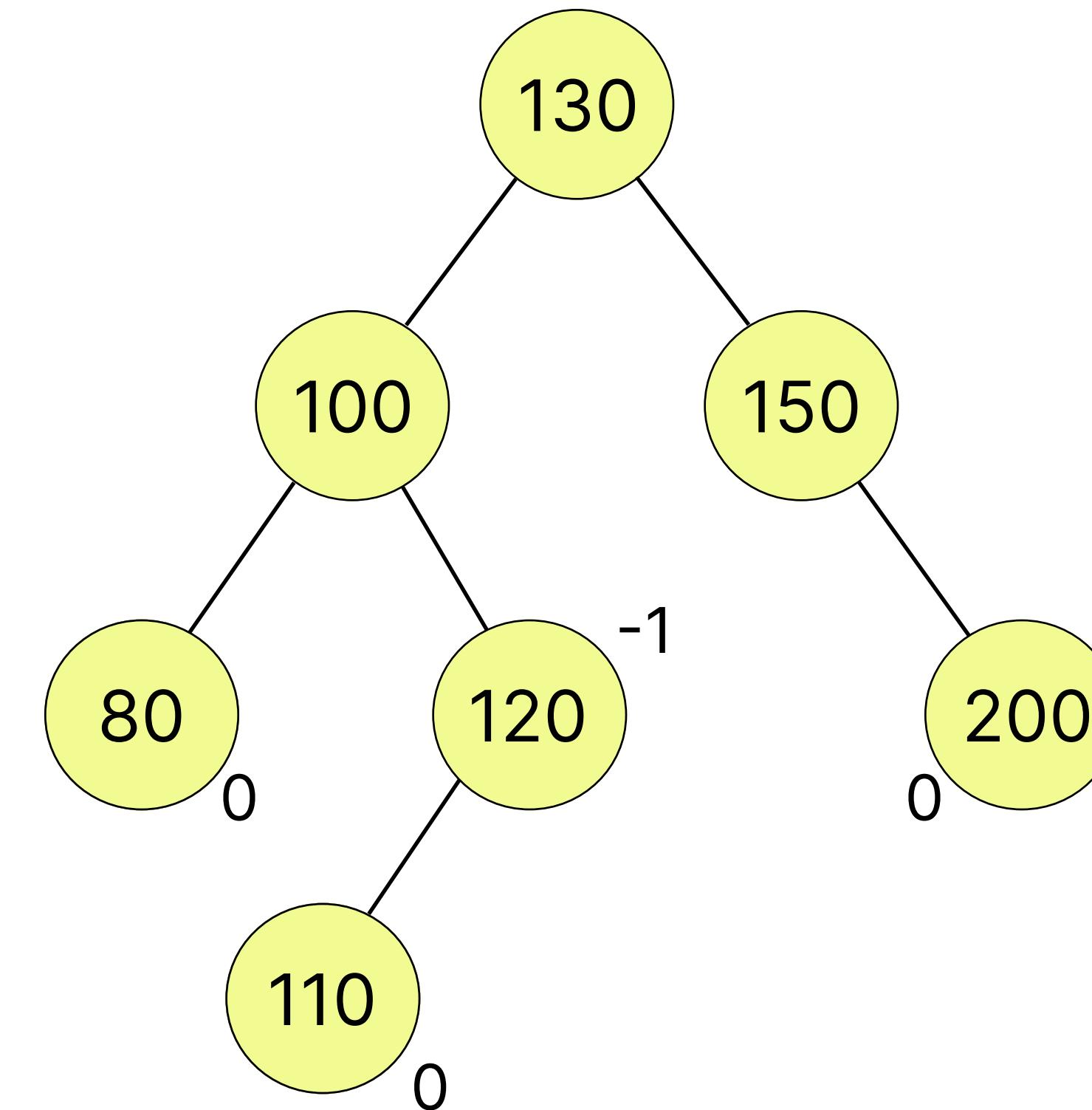
$$FB(n) = \text{altura}(no \rightarrow \text{dir}) - \text{altura}(no \rightarrow \text{esq})$$



Fator de Balanceamento (FB)

Fator de balanceamento: diferença entre altura da subárvore direita e esquerda

$$FB(n) = \text{altura}(no \rightarrow \text{dir}) - \text{altura}(no \rightarrow \text{esq})$$

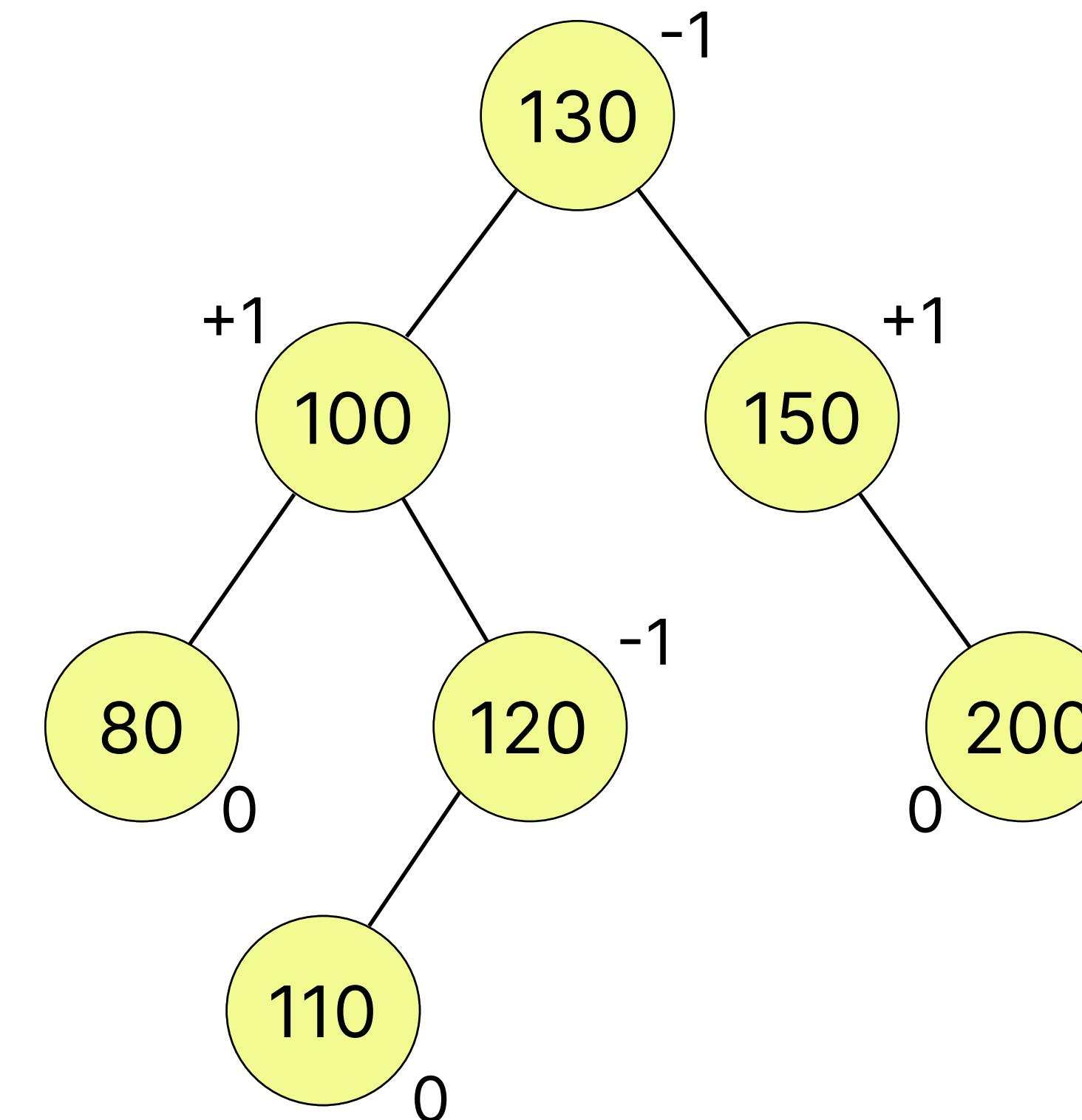


Uma árvore binária de busca é uma **AVL** quando, para qualquer um de seus nós, a diferença entre as alturas de suas subárvore **direita** e **esquerda** é no **máximo 1**

Fator de Balanceamento (FB)

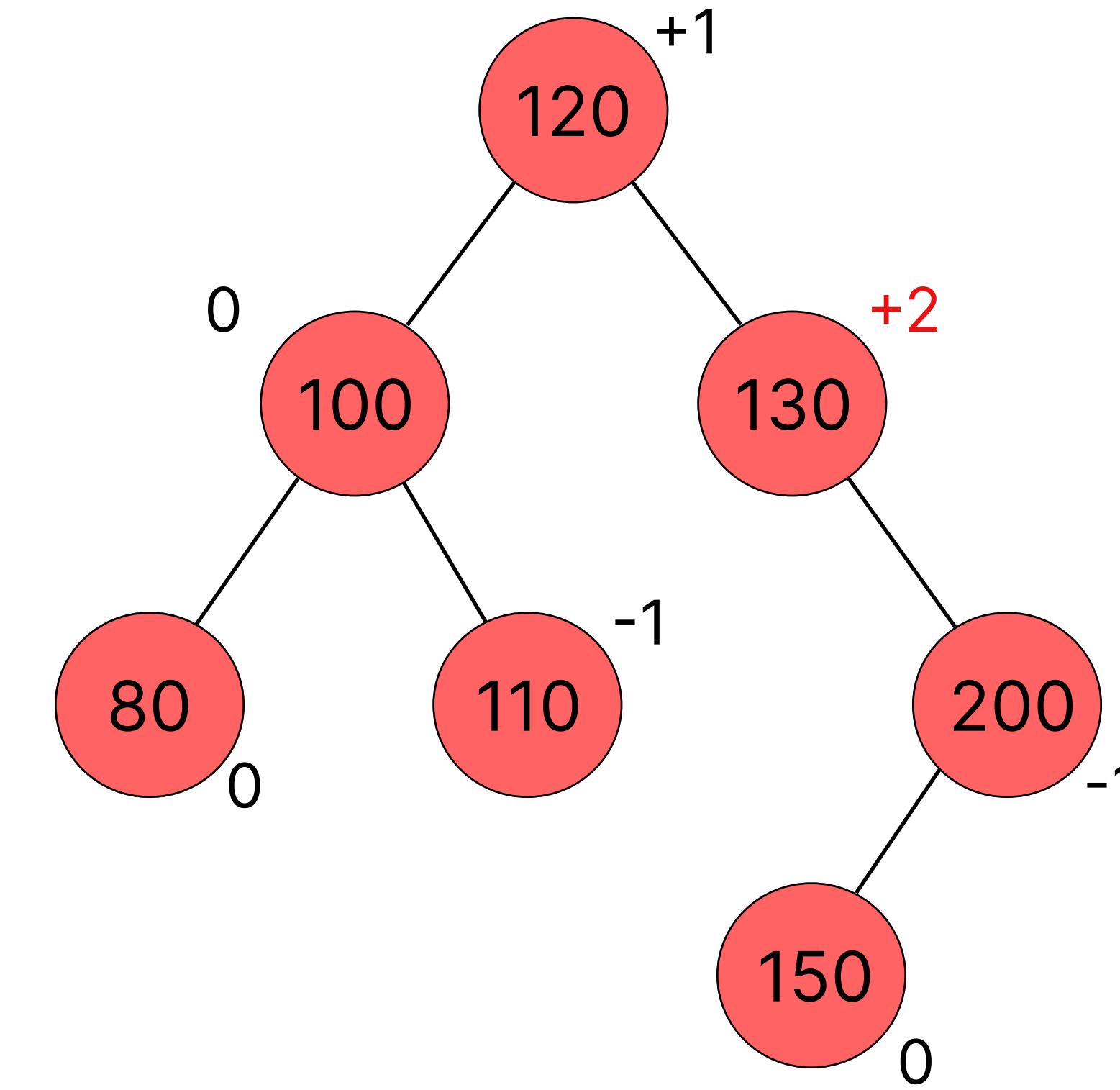
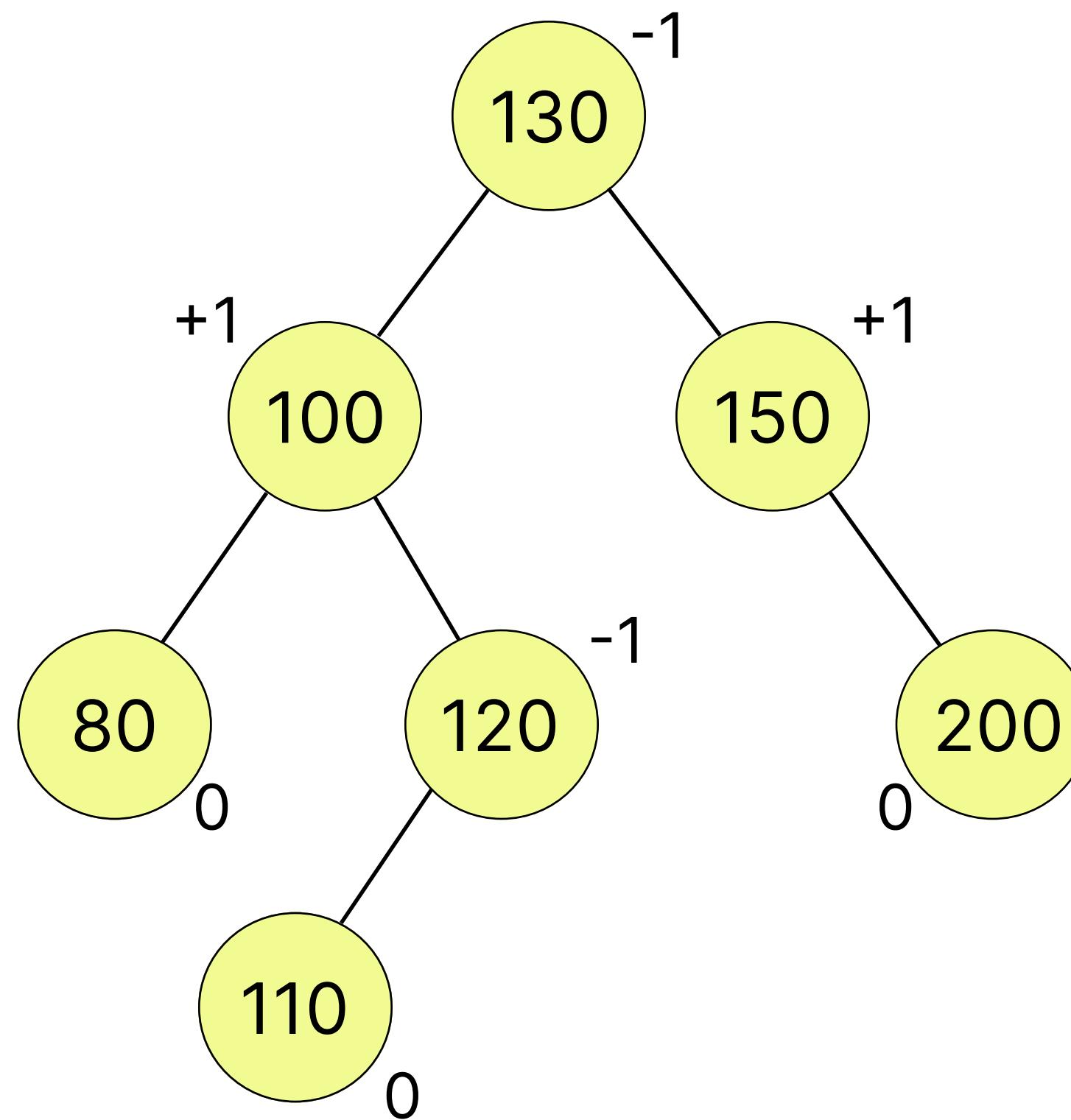
Fator de balanceamento: diferença entre altura da subárvore direita e esquerda

$$FB(n) = \text{altura}(no \rightarrow \text{dir}) - \text{altura}(no \rightarrow \text{esq})$$



Fator de Balanceamento (FB)

FB precisa ser **-1**, **0** ou **+1** em todos os nós da árvore para que seja uma **AVL**.



Balanceamento de árvores **AVL** por rotação

Quando uma inserção ou exclusão faz com que a árvore perca as propriedades AVL, deve-se realizar uma operação de reestruturação chamada Rotação

Tipos de rotação:

Rotação Simples

Direita

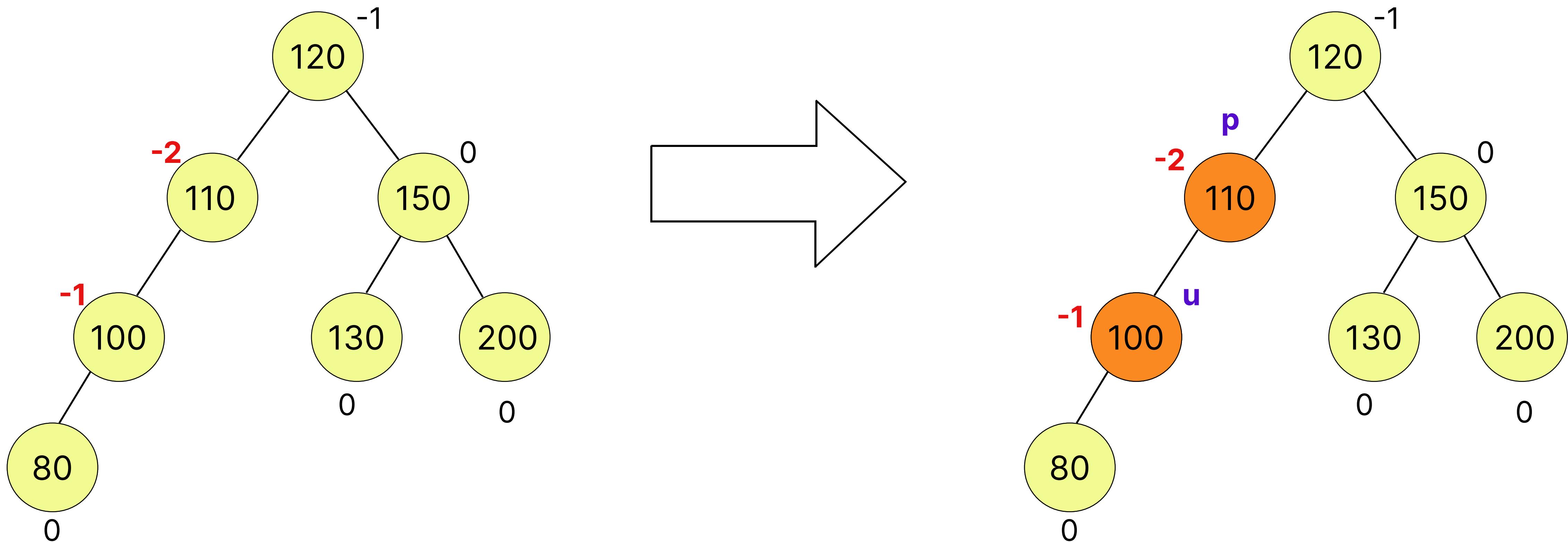
Esquerda

Rotação Dupla

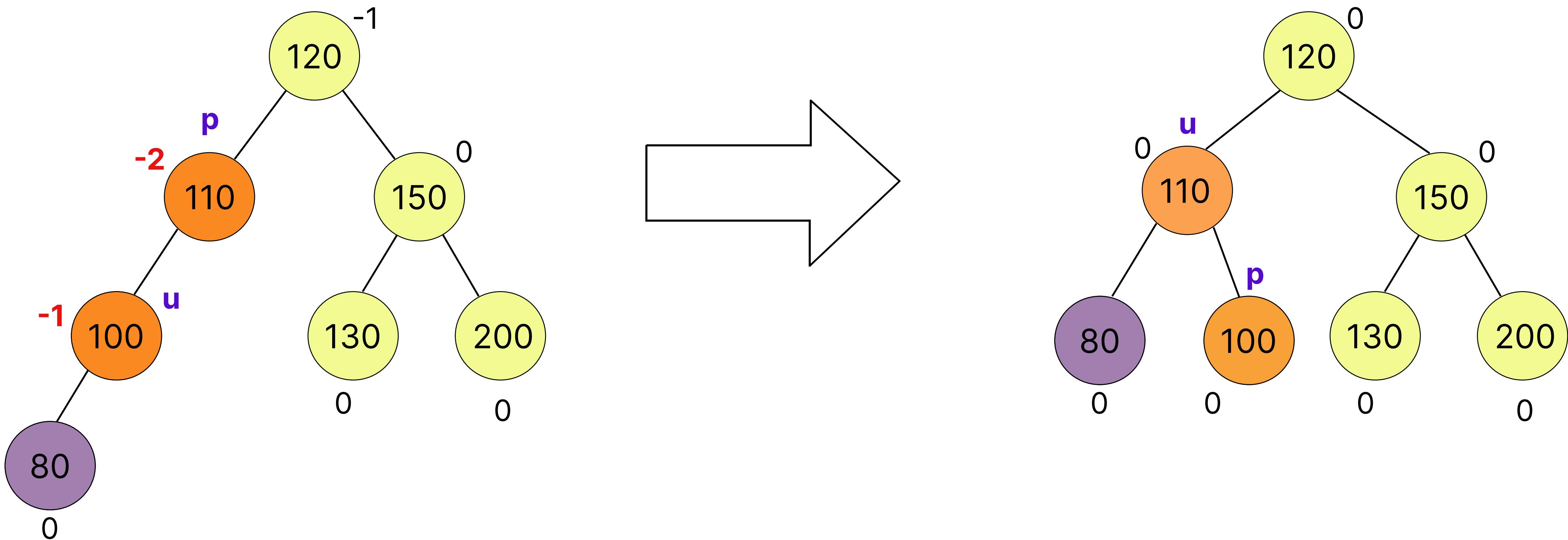
Direita (esquerda - direita)

Esquerda (direita - esquerda)

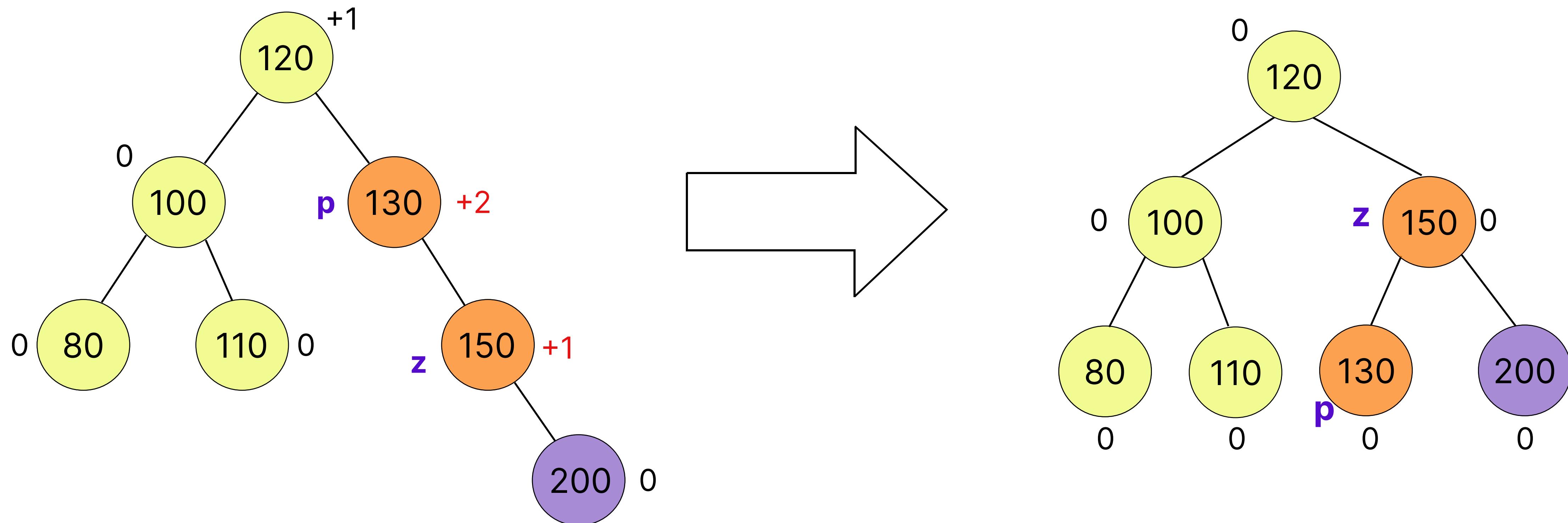
Exemplo: rotação direita



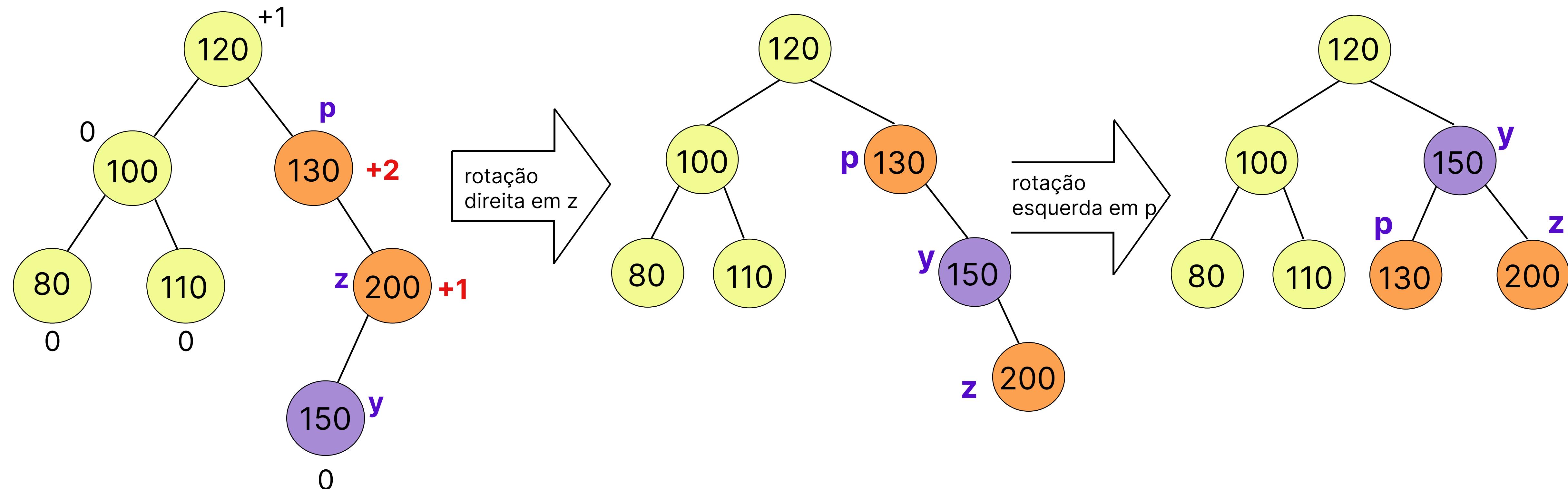
Exemplo: rotação direita



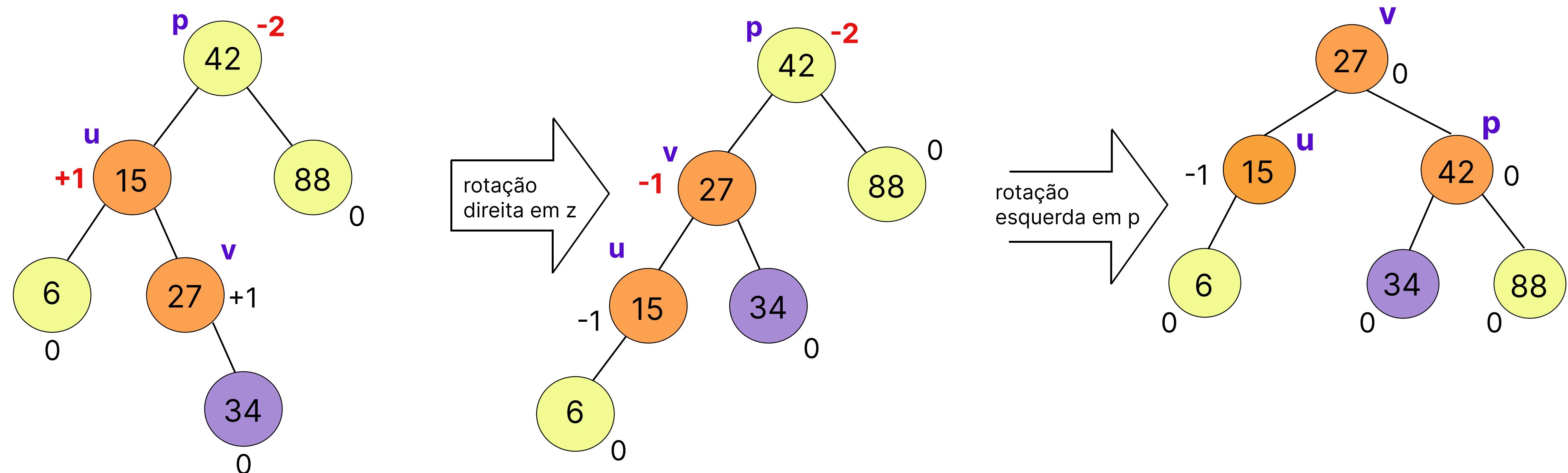
Exemplo: rotação esquerda



Exemplo: rotação dupla esquerda



Exemplo: rotação dupla direita



Quando aplicar

Nó com FB = -2 e filho com FB = -1 ou 0:

→ rotação do nó com FB = -2 p/direita

Nó com FB = +2 e filho com FB = +1 ou 0:

→ rotação do nó com FB = +2 p/esquerda

Sinais iguais: rotação simples

Nó com FB = -2 e filho com FB = +1:

→ rotação do nó com FB = +1 p/esquerda, e

→ rotação do nó com FB = -2 p/direita

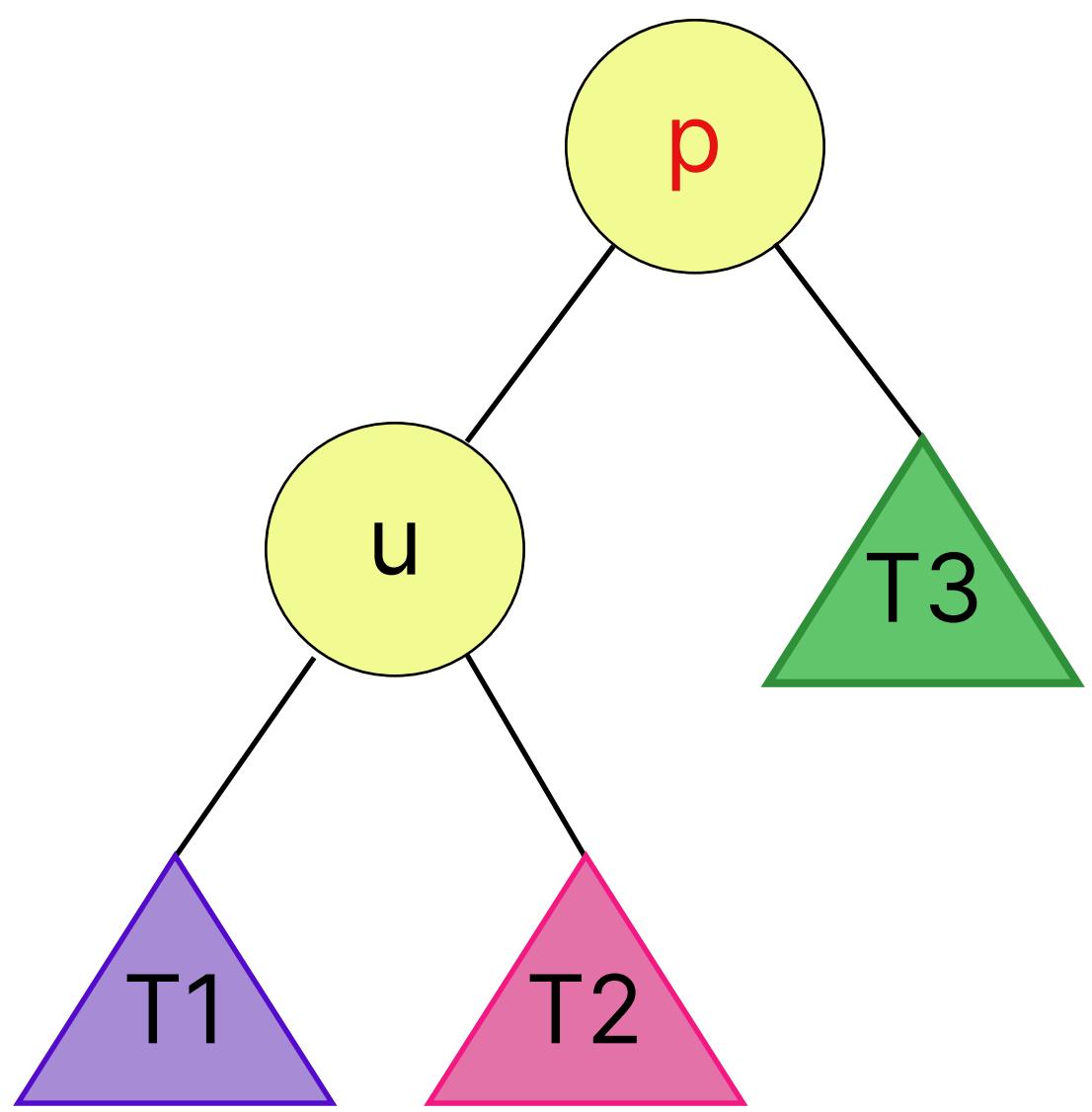
Nó com FB = +2 e filho com FB = -1:

→ rotação do nó com FB = -1 p/direita, e

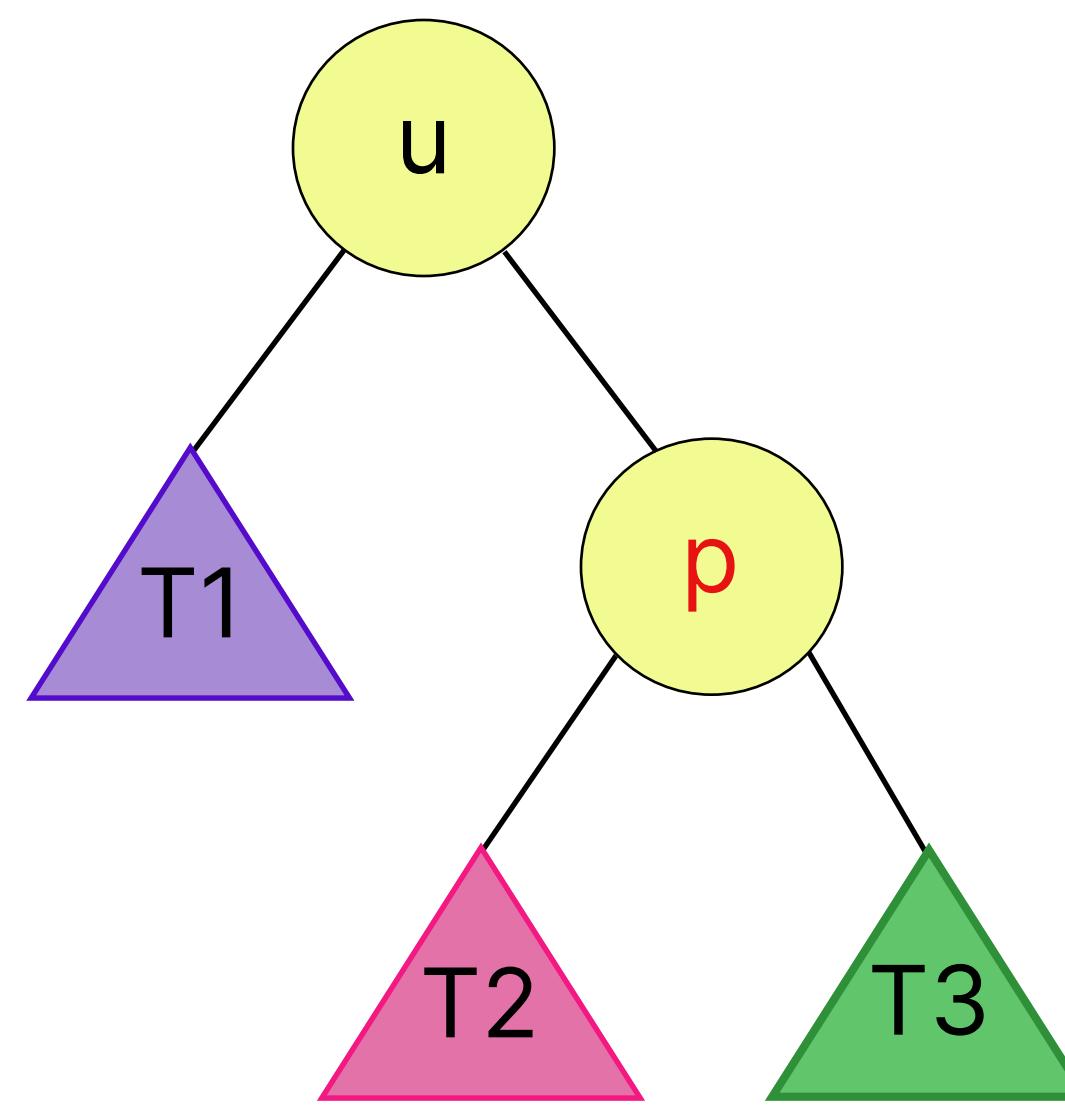
→ rotação do nó com FB = +2 p/esquerda

Sinais opostos: rotação dupla

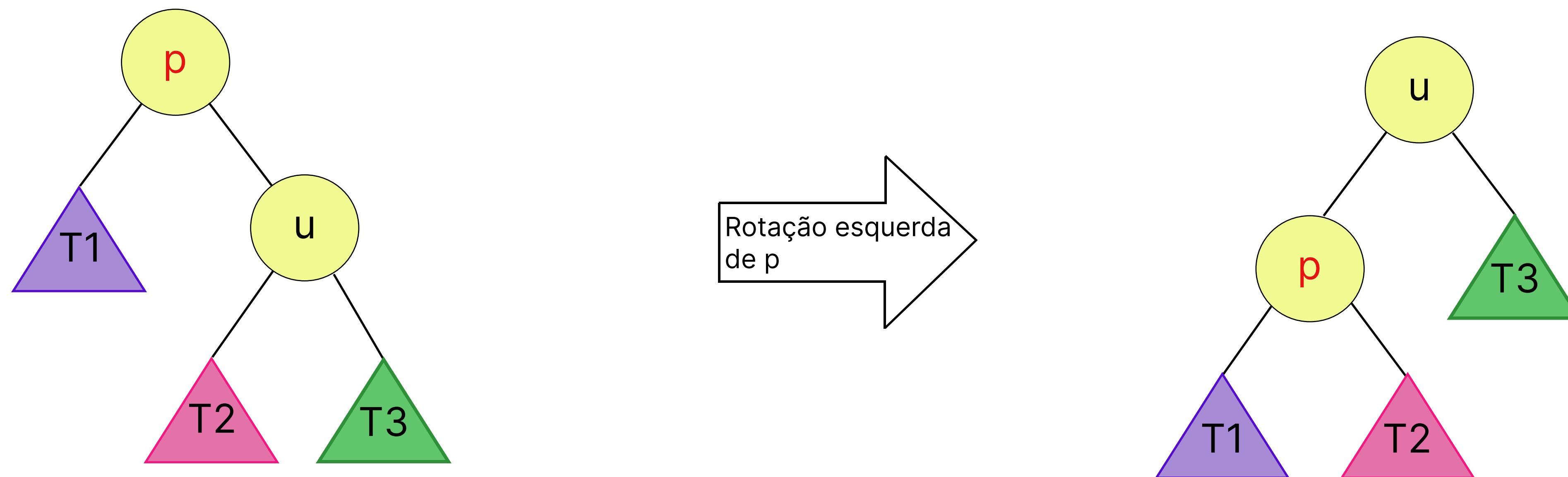
Rotação direita



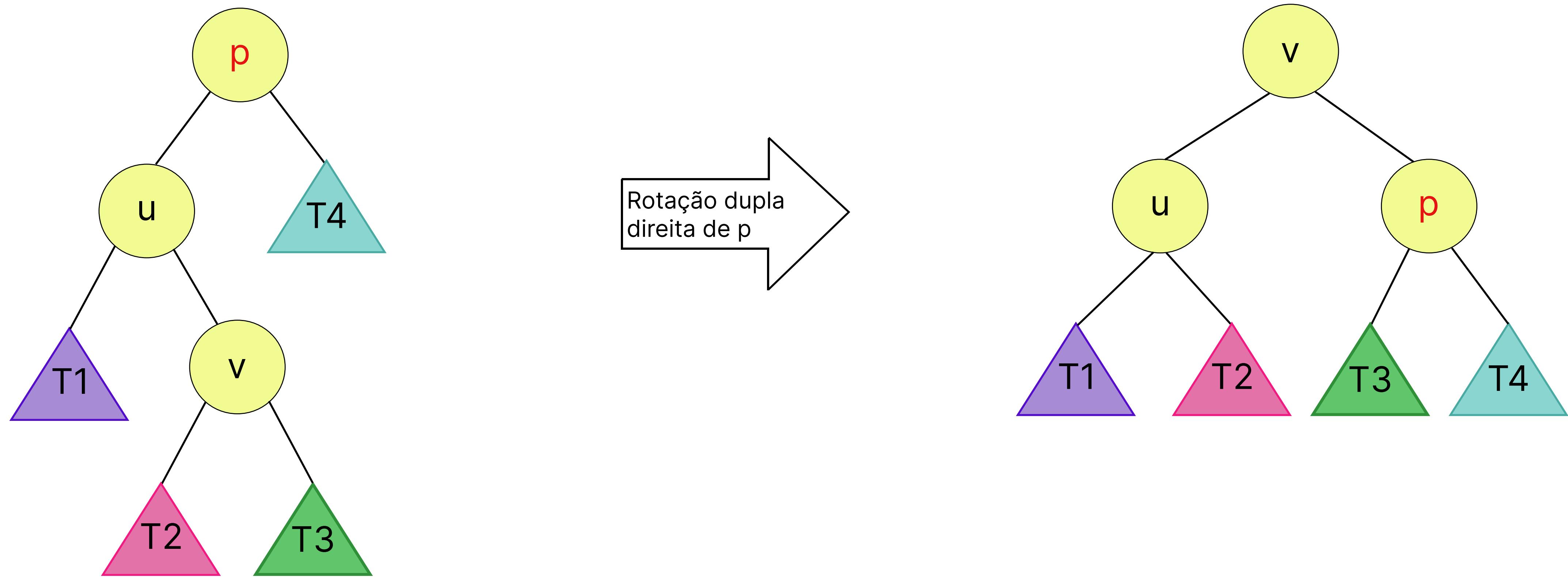
Rotação direita
de p



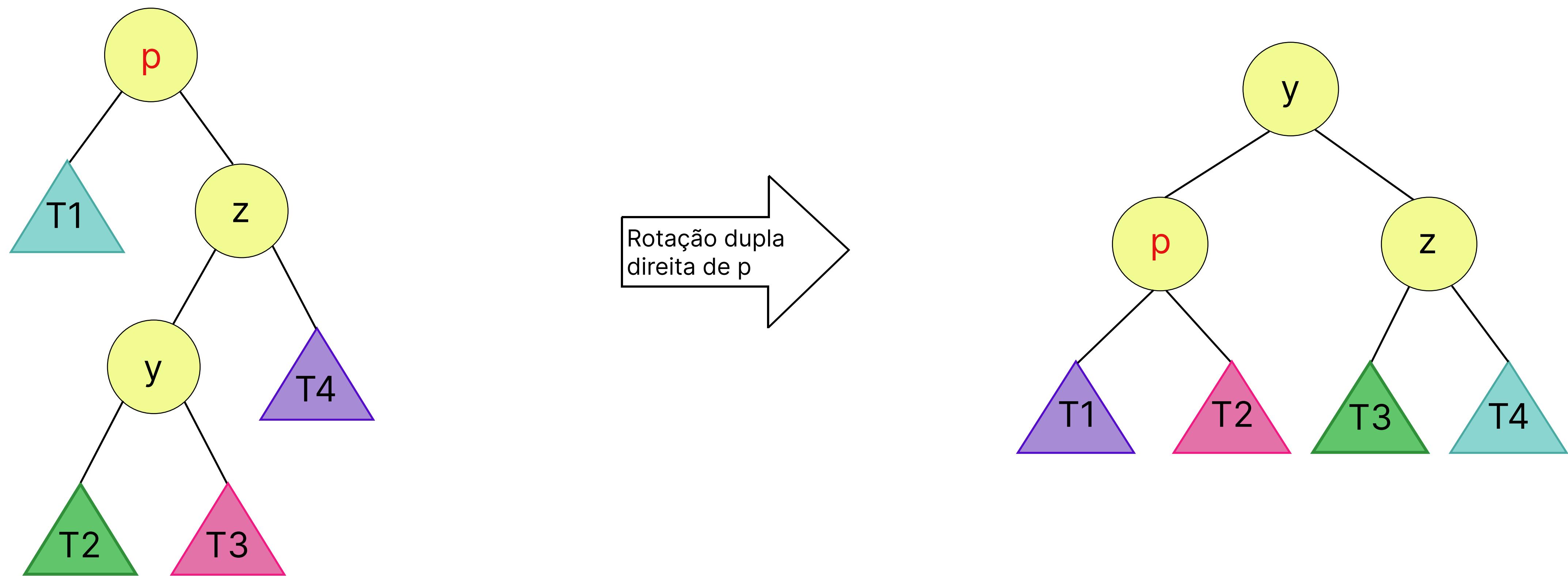
Rotação esquerda



Rotação dupla direita (esquerda - direita)



Rotação dupla esquerda (direita - esquerda)



Caso de teste com N = 1000

```
| 0- Sair;  
| 1- Inserir;  
| 2- Listar;  
| 3- Inserir numeros aleatorios;  
-----  
  
| Opciao:  
3  
Digite a quantidade de numeros aleatorios:1000  
Tempo de execucao: 0
```

Caso de teste com N = 100

```
No = 24, altura = 6, FATBAL = 1  
No = 16, altura = 5, FATBAL = 1  
No = 4, altura = 4, FATBAL = 0  
No = 2, altura = 3, FATBAL = 1  
No = 0, altura = 2, FATBAL = -1  
No = 1, altura = 1, FATBAL = 0  
No = 3, altura = 1, FATBAL = 0  
No = 11, altura = 3, FATBAL = 1  
No = 6, altura = 2, FATBAL = 0  
No = 5, altura = 1, FATBAL = 0  
No = 8, altura = 1, FATBAL = 0  
No = 12, altura = 1, FATBAL = 0  
No = 21, altura = 3, FATBAL = -1  
No = 18, altura = 1, FATBAL = 0  
No = 22, altura = 2, FATBAL = -1  
No = 23, altura = 1, FATBAL = 0  
No = 34, altura = 4, FATBAL = 0  
No = 27, altura = 3, FATBAL = -1  
No = 26, altura = 1, FATBAL = 0  
No = 31, altura = 2, FATBAL = 0  
No = 29, altura = 1, FATBAL = 0  
No = 33, altura = 1, FATBAL = 0  
No = 38, altura = 3, FATBAL = 0  
No = 36, altura = 2, FATBAL = 0  
No = 35, altura = 1, FATBAL = 0  
No = 37, altura = 1, FATBAL = 0
```

```
No = 39, altura = 1, FATBAL = 0  
No = 67, altura = 6, FATBAL = 0  
No = 58, altura = 5, FATBAL = 1  
No = 48, altura = 4, FATBAL = 0  
No = 45, altura = 3, FATBAL = 0  
No = 42, altura = 2, FATBAL = -1  
No = 44, altura = 1, FATBAL = 0  
No = 47, altura = 2, FATBAL = 1  
No = 46, altura = 1, FATBAL = 0  
No = 53, altura = 3, FATBAL = -1  
No = 50, altura = 1, FATBAL = 0  
No = 56, altura = 2, FATBAL = 0  
No = 54, altura = 1, FATBAL = 0  
No = 57, altura = 1, FATBAL = 0  
No = 62, altura = 3, FATBAL = 0  
No = 61, altura = 2, FATBAL = 1  
No = 59, altura = 1, FATBAL = 0  
No = 64, altura = 2, FATBAL = -1  
No = 66, altura = 1, FATBAL = 0  
No = 91, altura = 5, FATBAL = 1  
No = 78, altura = 4, FATBAL = 0  
No = 71, altura = 3, FATBAL = 0  
No = 69, altura = 2, FATBAL = 0  
No = 68, altura = 1, FATBAL = 0  
No = 70, altura = 1, FATBAL = 0  
No = 73, altura = 2, FATBAL = -1  
No = 76, altura = 1, FATBAL = 0  
No = 82, altura = 3, FATBAL = -1
```

```
No = 81, altura = 1, FATBAL = 0  
No = 88, altura = 2, FATBAL = 0  
No = 84, altura = 1, FATBAL = 0  
No = 90, altura = 1, FATBAL = 0  
No = 95, altura = 3, FATBAL = 1  
No = 93, altura = 2, FATBAL = 0  
No = 92, altura = 1, FATBAL = 0  
No = 94, altura = 1, FATBAL = 0  
No = 99, altura = 1, FATBAL = 0
```



```
| 0- Sair;  
| 1- Inserir;  
| 2- Listar;  
| 3- Inserir numeros aleatorios;  
-----  
| Opcão:  
3  
Digite a quantidade de numeros aleatorios:100  
Tempo de execucao: 0
```

Caso de teste com N = 10

```
No = 4, altura = 3, FATBAL = 0  
No = 1, altura = 2, FATBAL = 0  
No = 0, altura = 1, FATBAL = 0  
No = 2, altura = 1, FATBAL = 0  
No = 8, altura = 2, FATBAL = 0  
No = 7, altura = 1, FATBAL = 0  
No = 9, altura = 1, FATBAL = 0
```

```
| 0- Sair;  
| 1- Inserir;  
| 2- Listar;  
| 3- Inserir numeros aleatorios;  
-----  
| Opcao:  
3  
Digite a quantidade de numeros aleatorios:10  
Tempo de execucao: 0
```

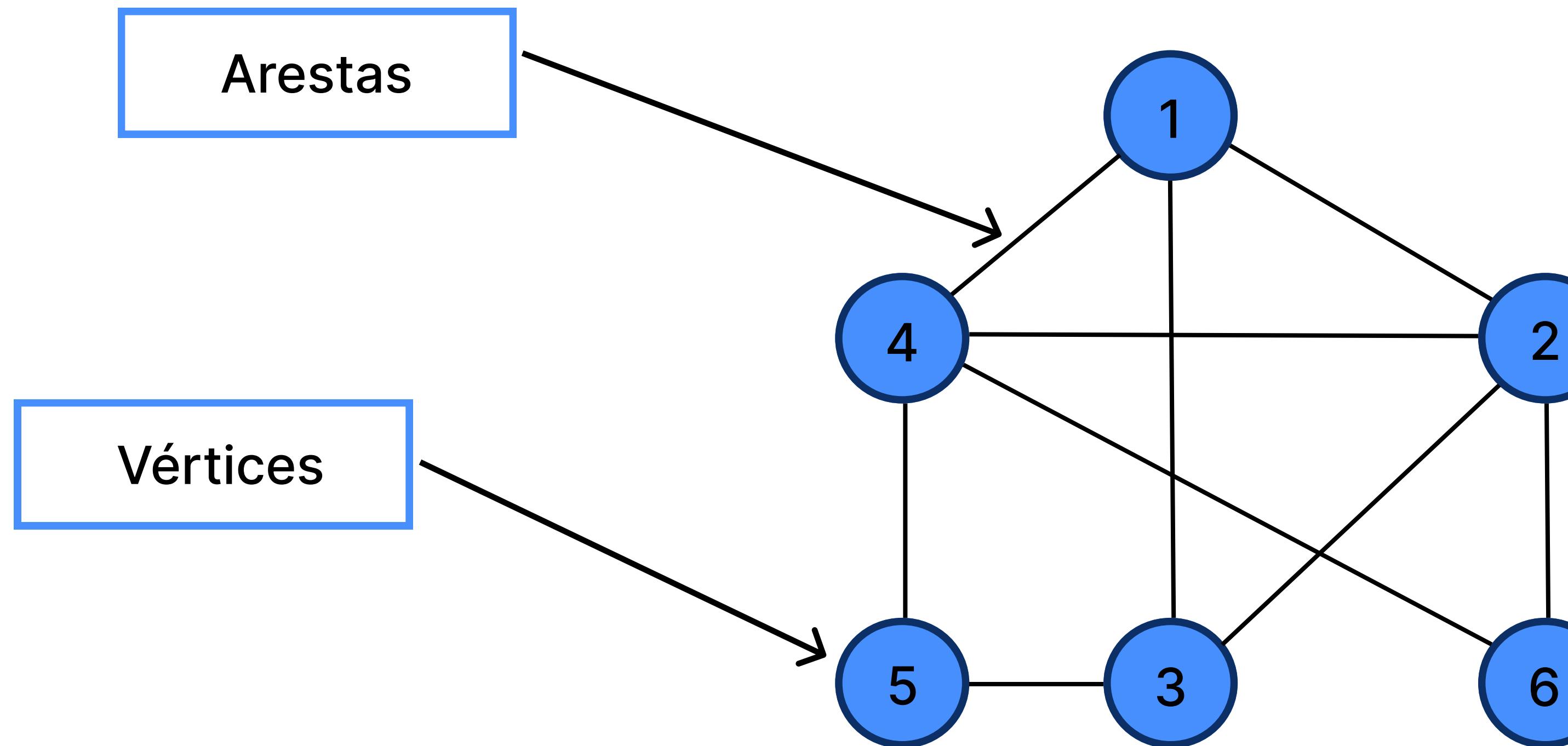
Caso de teste com N = 1

```
No = 0, altura = 1, FATBAL = 0
```

```
| 0- Sair;  
| 1- Inserir;  
| 2- Listar;  
| 3- Inserir numeros aleatorios;  
-----  
  
| Opcão:  
3  
Digite a quantidade de numeros aleatorios:1  
Tempo de execucao: 0
```

Grafos

Grafos são estruturas de dados formadas por um conjunto de vértices e um conjunto de arestas

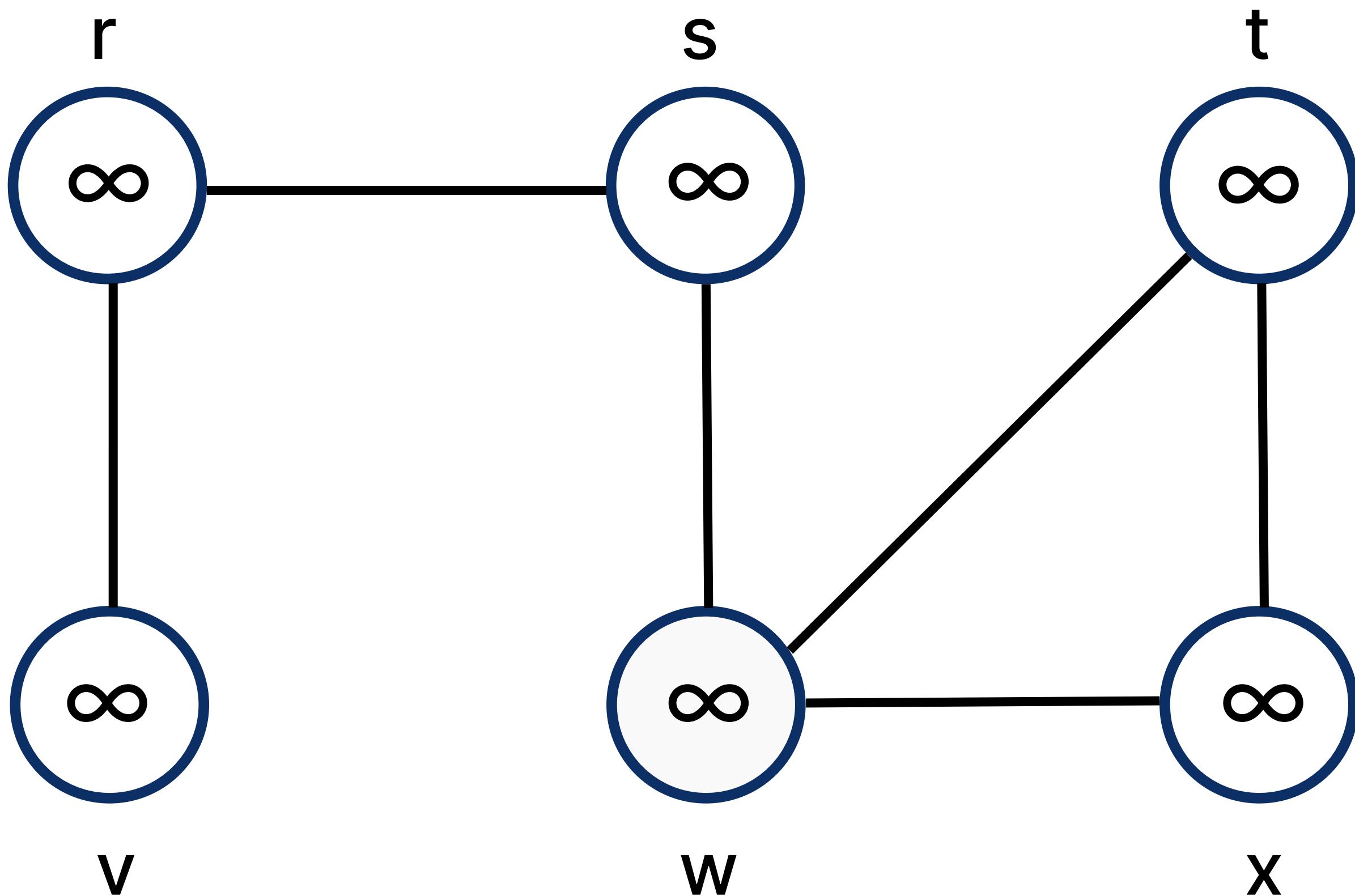


Busca em largura

A **busca em largura** é um algoritmo de busca em grafos utilizado para realizar uma busca ou travessia num grafo ou uma árvore

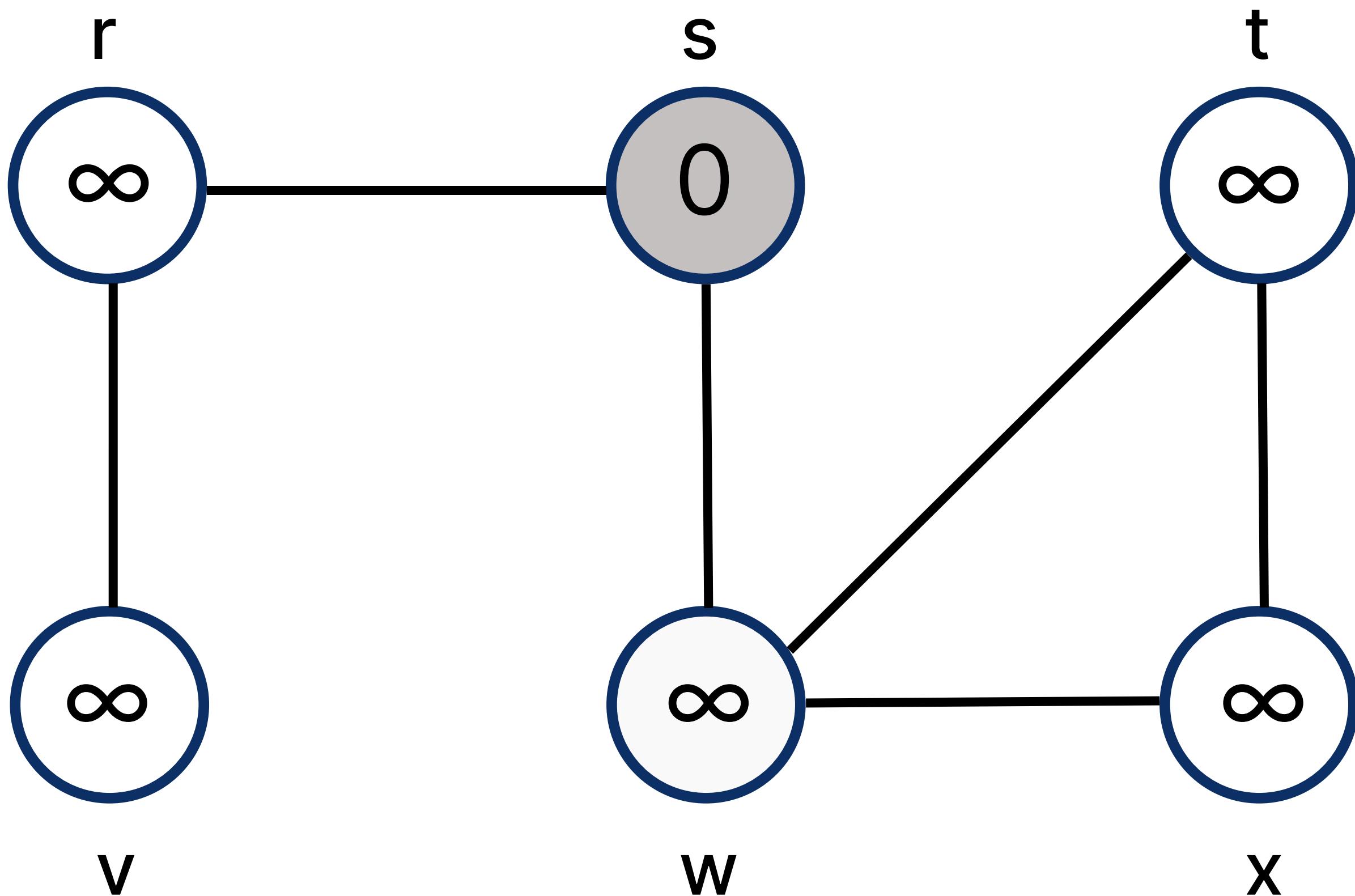
O algoritmo **numera** os vértices, em sequência, na ordem em que eles são descobertos. Para fazer isso, o algoritmo usa uma **fila** de vértices. No começo de cada iteração, a fila contém vértices que já foram numerados mas têm vizinhos ainda não numerados.

Exemplo



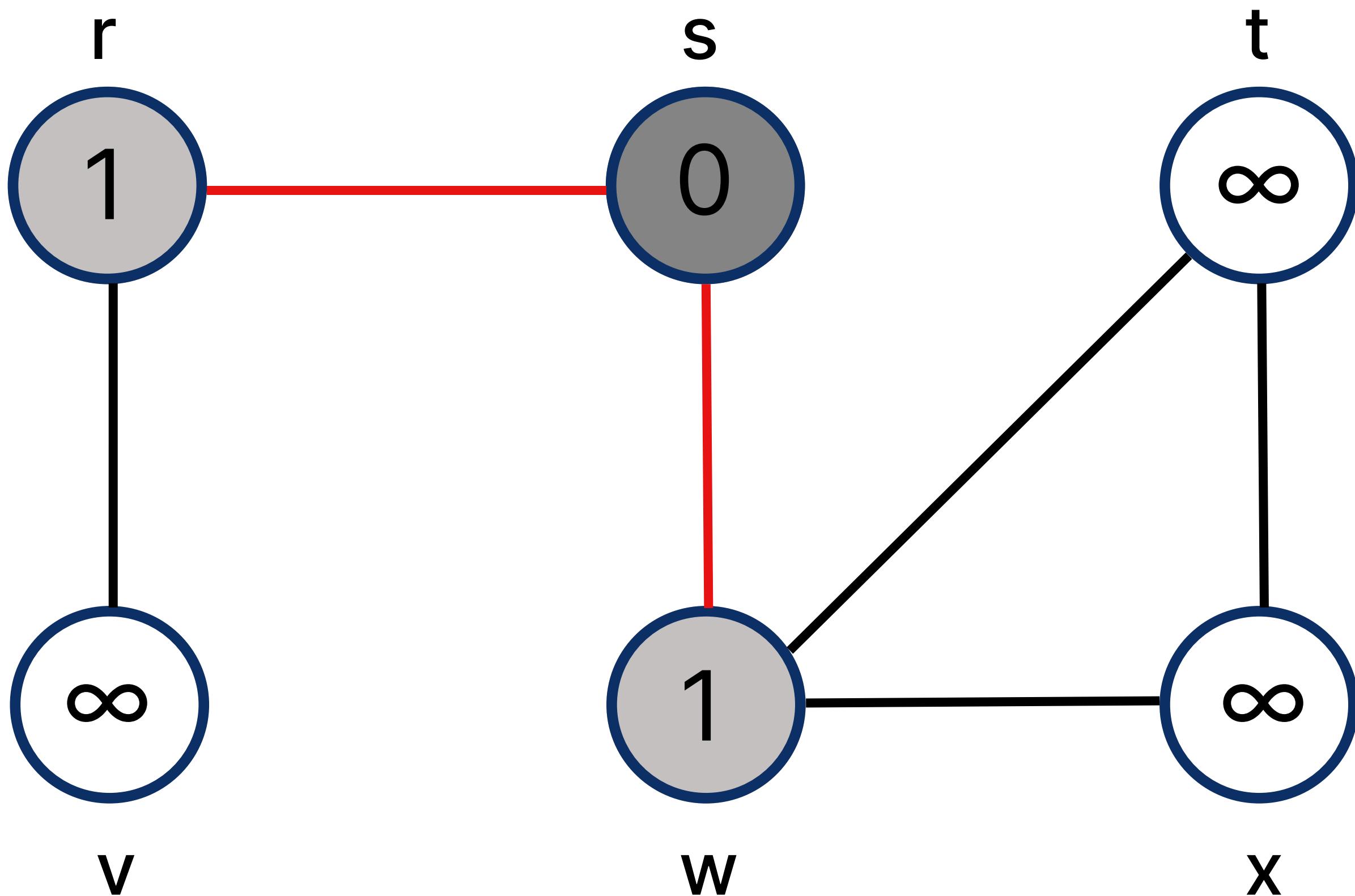
Fila:

Exemplo



Fila: s

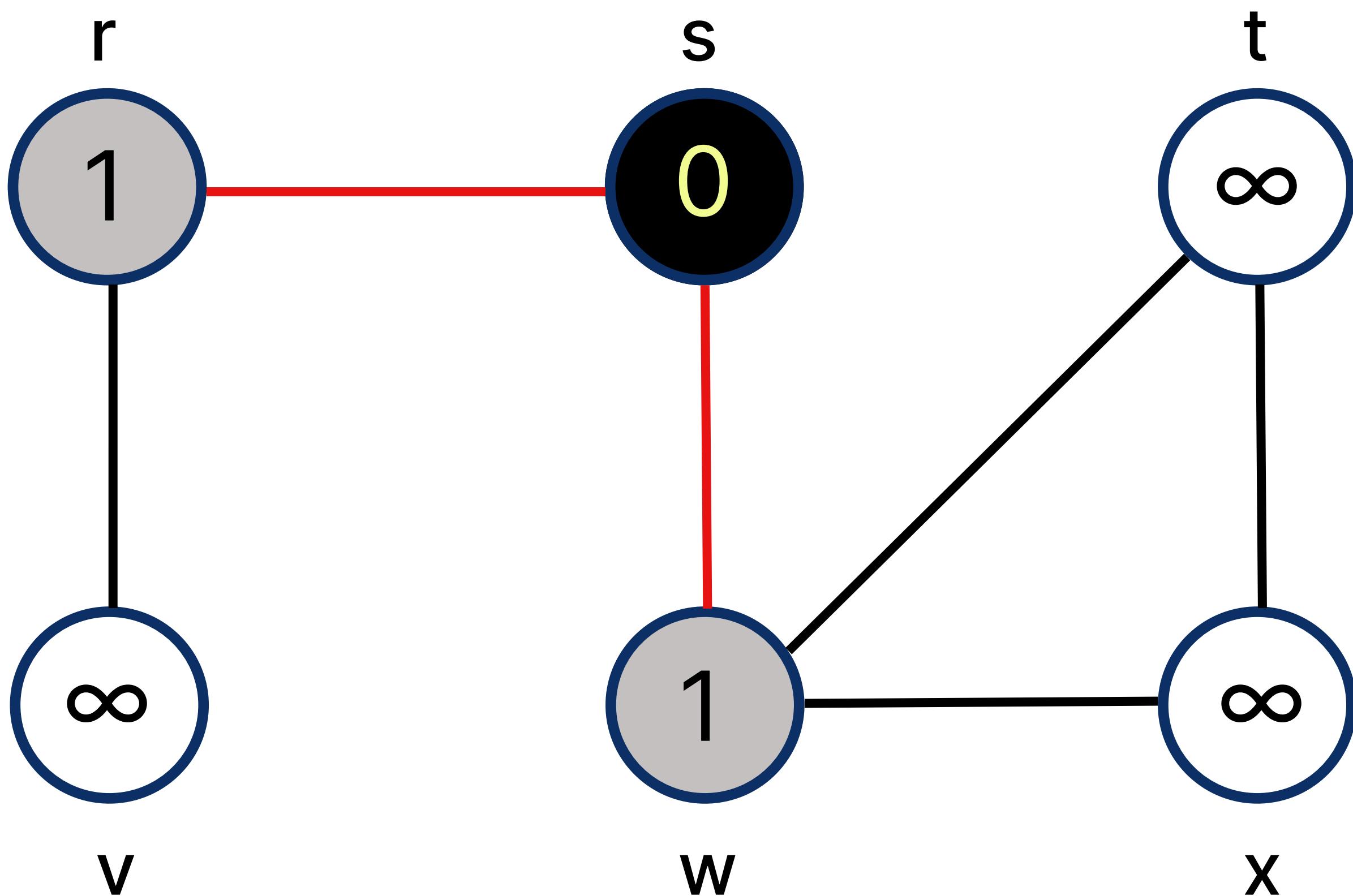
Exemplo



Fila:

w	r
---	---

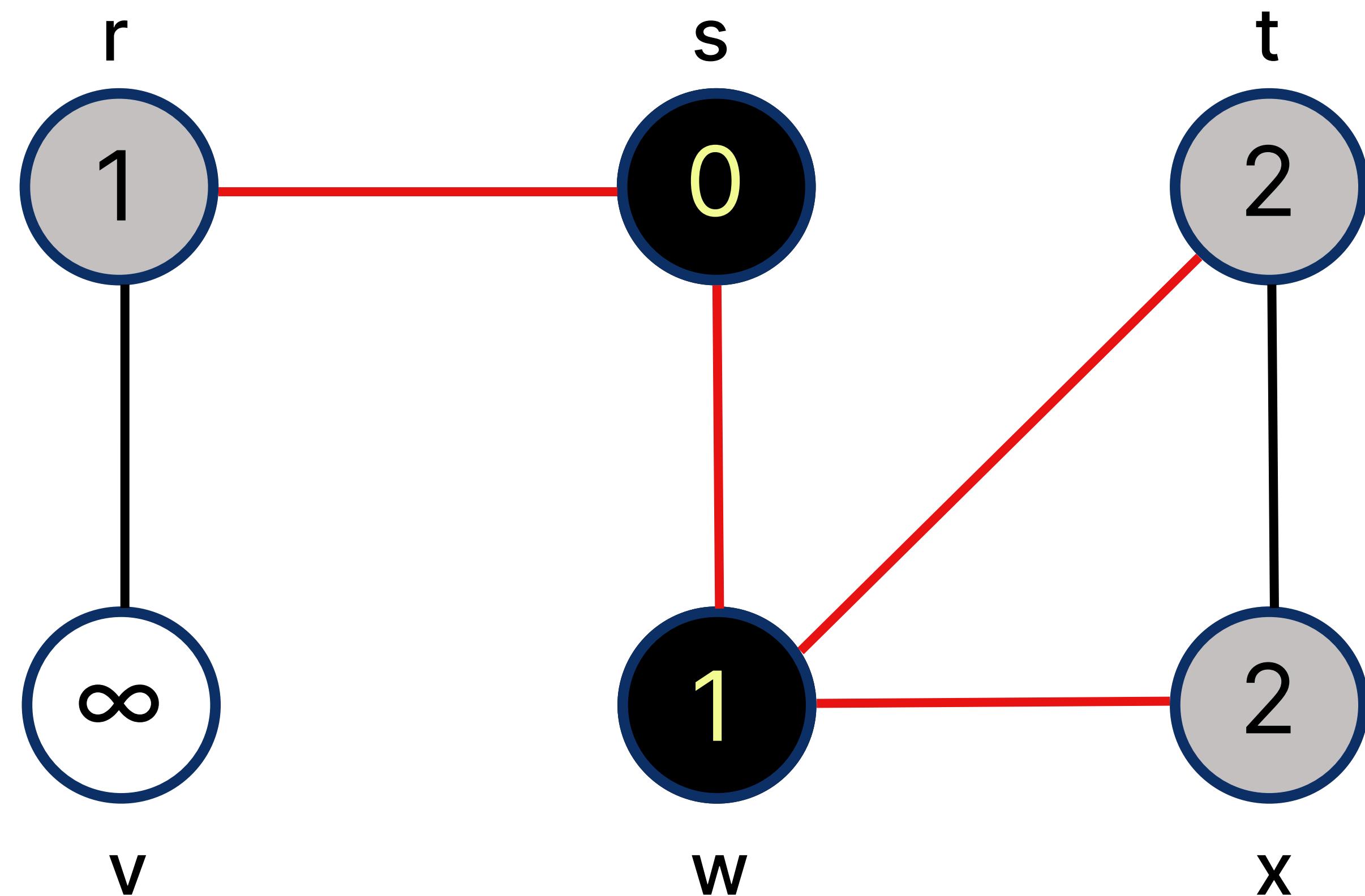
Exemplo



Fila:

w	r
---	---

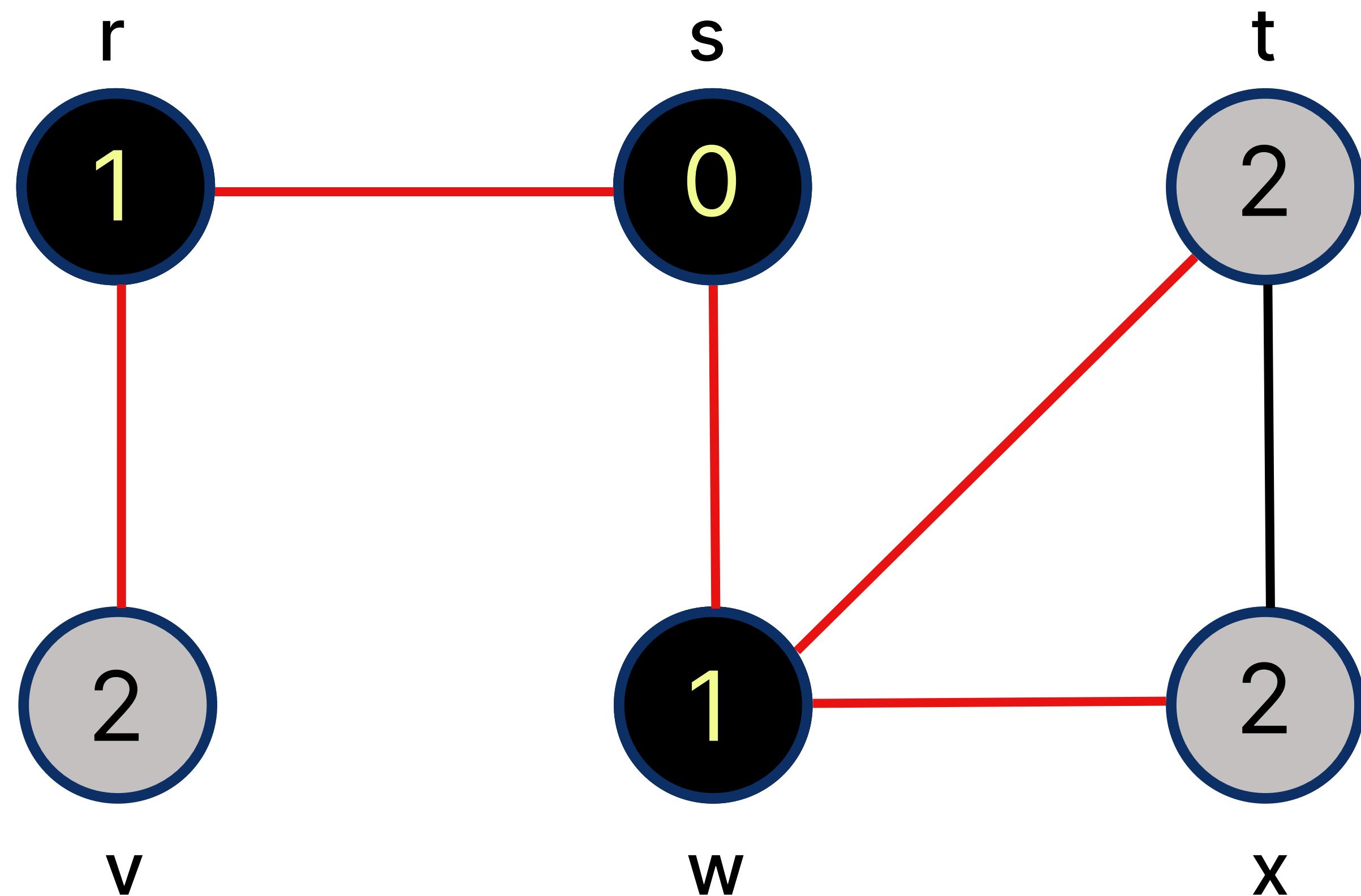
Exemplo



Fila:

r	t	x
-----	-----	-----

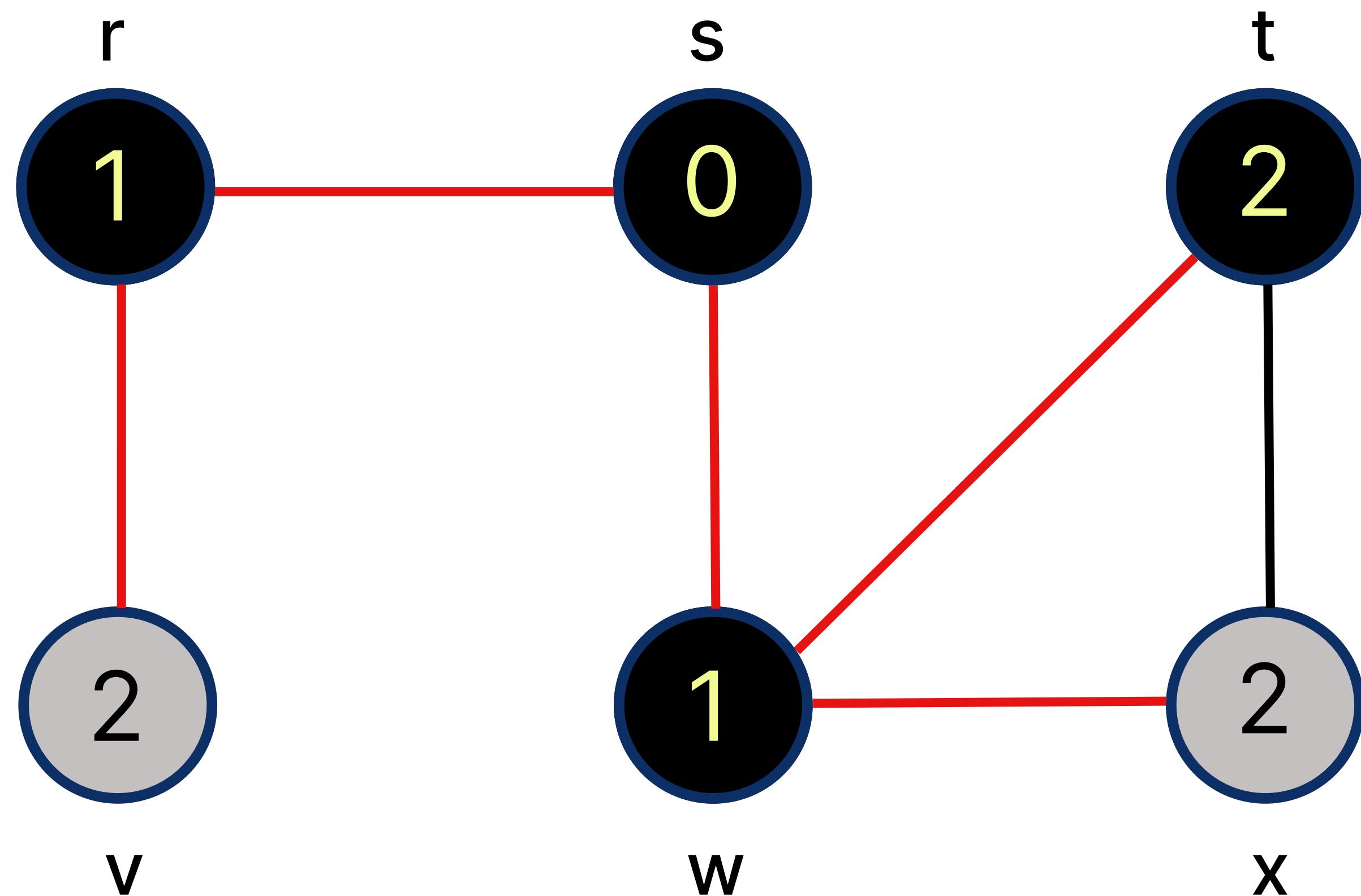
Exemplo



Fila:

t	x	v
---	---	---

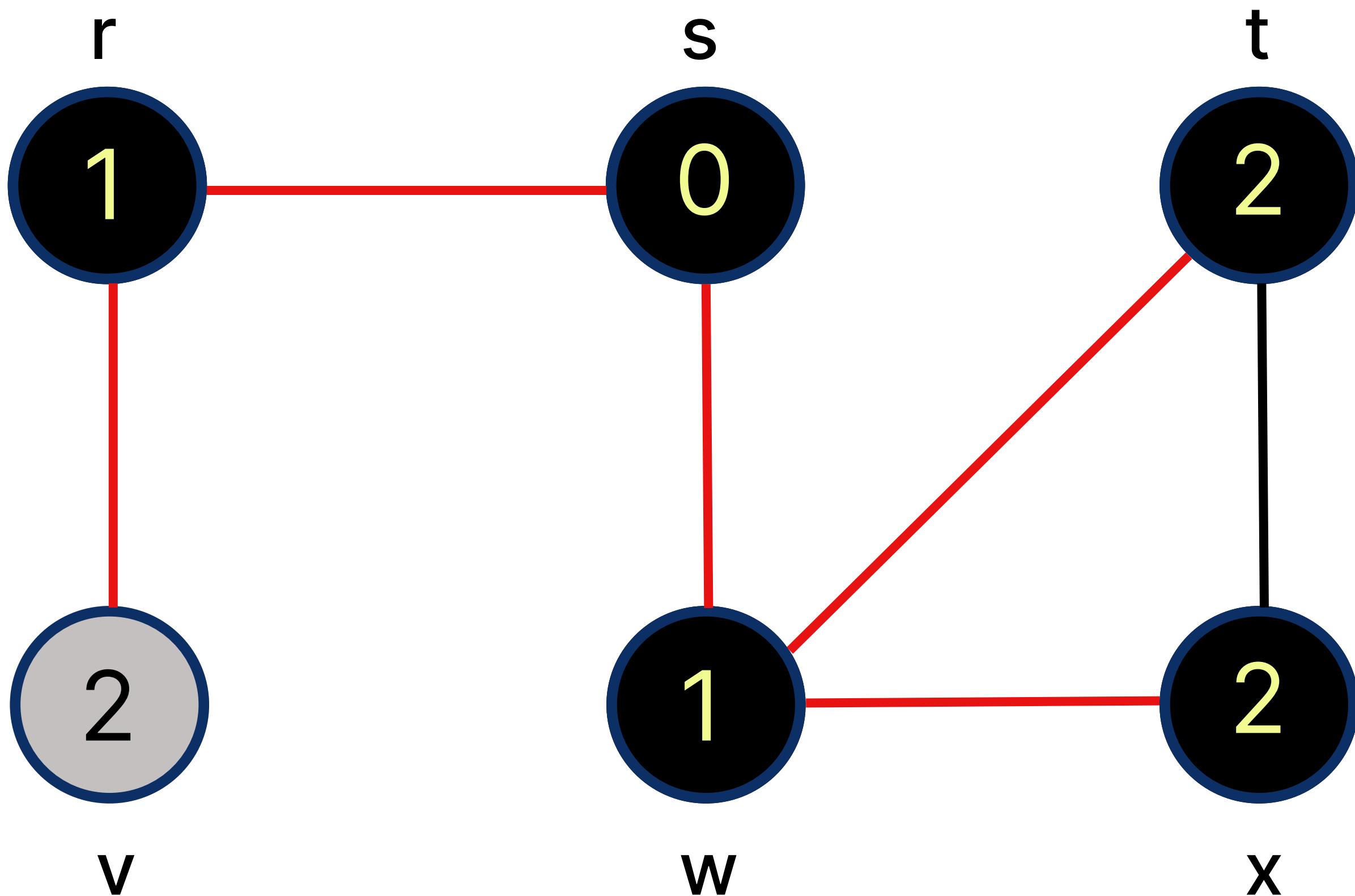
Exemplo



Fila:

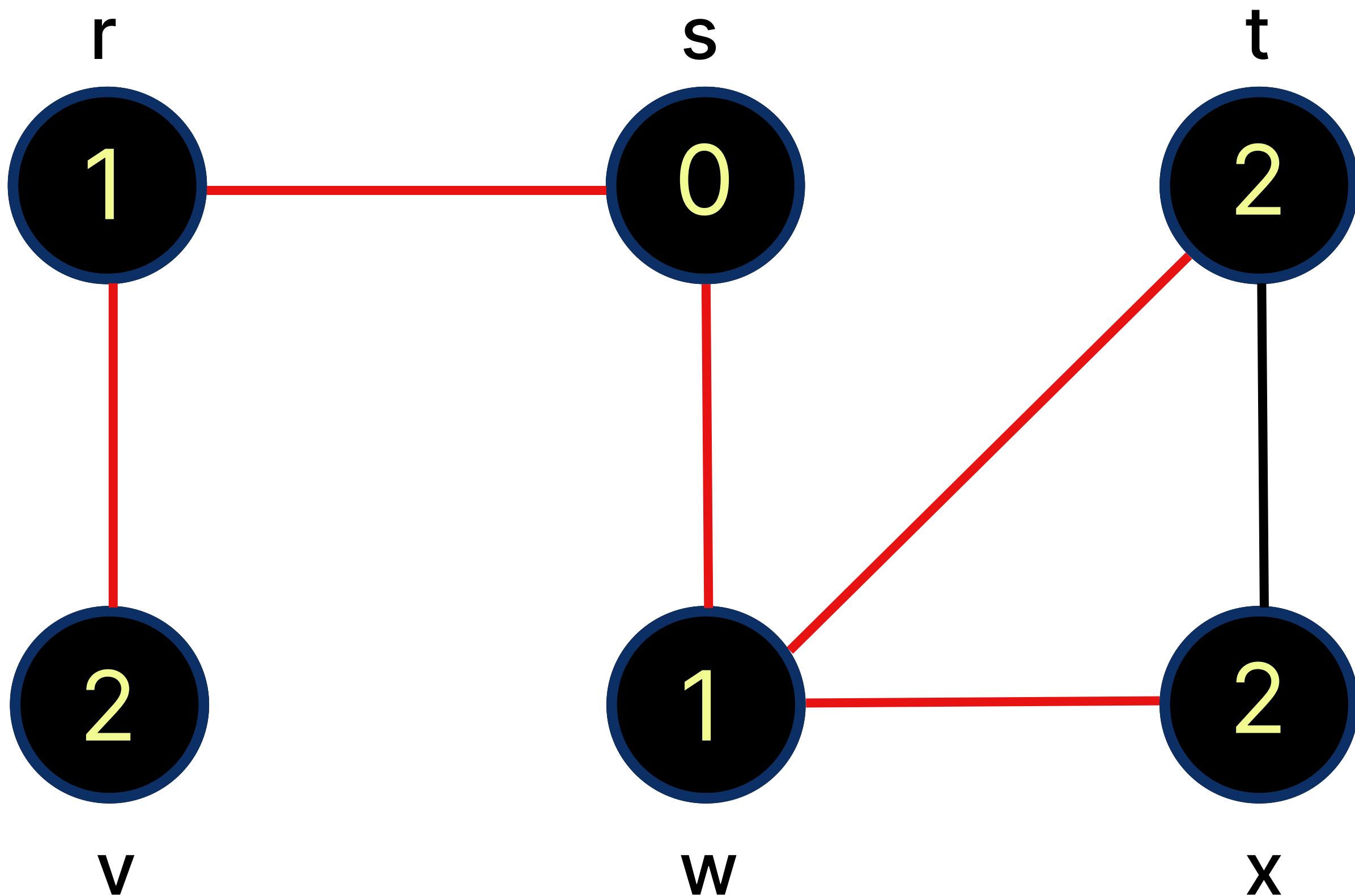
x	v
---	---

Exemplo



Fila:

Exemplo



Fila:

Busca em largura

Complexidade de tempo:

aquele em que todos os vértices e arestas são explorados pelo algoritmo. Dada pela expressão $O(|A| + |v|)$.

Complexidade de Espaço:

Quando o número de vértices no grafo é conhecido e supondo-se a representação deste em listas de adjacência, a complexidade de espaço do algoritmo pode ser representada por $O(|v|)$ onde $|v|$ representa o número total de vértices.

Busca em largura de N = 1000

```
1000 54 963 186 966 886 558 549 903 465 509 39 338 72 477 484 394 256 874 562 154 906 19 81 962 246 389 180 624 5 279 27
2 8 809 975 382 539 608 649 739 637 424 628 233 296 521 441 788 369 777 169 837 869 594 687 201 200 492 596 630 808 800
946 153 814 433 730 366 913 557 414 365 322 293 538 772 371 813 585 265 162 239 532 528 130 140 626 214 819 941 533 693
482 907 56 826 292 254 679 506 341 892 779 478 449 980 487 786 349 713 42 917 677 854 929 568 603 422 807 782 833 910 37
4 997 88 120 955 175 515 648 4 891 875 76 470 429 870 85 550 994 217 520 1 642 559 601 998 440 383 543 625 364 12 351 15
2 627 94 388 516 735 29 498 972 362 182 261 590 525 238 934 241 203 861 318 993 611 852 964 198 368 13 145 16 273 460 84
75 778 25 927 179 547 425 136 77 488 530 504 109 415 353 579 667 325 205 197 546 586 468 297 904 98 493 763 319 446 406
717 269 965 933 644 280 801 83 889 820 930 459 248 57 792 335 914 787 671 461 682 938 143 96 28 211 497 438 435 791 385
80 451 986 811 970 511 250 673 959 59 945 165 696 302 856 724 707 253 334 73 151 685 935 358 324 609 376 344 445 270 19
4 456 282 185 110 544 621 841 278 865 755 288 62 184 699 799 24 518 490 798 117 635 615 496 743 524 829 485 107 897 137
447 618 842 132 566 432 894 574 61 741 263 595 227 298 789 173 688 719 605 830 315 370 207 79 160 781 956 784 223 999 40
9 806 3 542 128 674 881 181 513 968 746 656 581 859 727 871 171 541 694 427 343 18 560 466 535 883 702 119 816 662 619 9
9 106 393 501 620 264 7 529 616 475 50 463 224 937 622 704 760 583 942 925 802 536 193 600 979 678 423 155 641 320 392 1
92 494 580 183 701 290 715 113 983 742 866 868 323 336 949 876 527 602 569 776 563 657 480 74 196 726 22 86 828 985 431
434 757 653 680 512 848 458 395 304 973 578 976 419 436 473 663 599 765 44 588 728 775 669 873 289 23 668 681 631 758 31
1 231 115 722 785 420 37 230 166 862 354 729 146 652 9 770 951 36 417 901 244 969 245 754 896 491 312 271 825 832 450 37
3 163 287 537 206 284 860 882 416 672 952 387 665 249 442 209 650 575 797 517 448 222 316 147 310 872 607 89 589 157 268
638 523 242 87 391 92 381 556 831 2 759 505 780 102 610 774 634 909 839 812 277 125 725 851 455 606 905 407 190 947 170
53 974 108 218 960 857 286 167 356 718 982 329 647 453 176 584 228 78 706 526 486 32 923 700 597 711 361 437 967 64 314
195 257 991 116 761 822 655 751 790 540 898 939 129 133 623 651 377 764 213 769 159 300 977 748 11 43 817 810 65 659 68
9 551 987 259 114 225 471 317 895 845 714 592 252 795 105 900 384 926 564 33 38 301 587 971 709 666 352 664 397 576 399
691 617 34 348 142 332 187 944 426 210 936 327 827 948 149 502 234 101 328 996 172 483 768 697 577 405 177 824 359 636 7
49 111 124 421 834 188 444 686 41 855 712 753 150 258 402 723 676 112 978 291 571 189 330 988 294 303 640 45 888 412 794
950 705 911 199 411 510 499 639 698 863 573 30 737 508 612 452 31 877 82 943 695 591 479 912 68 899 815 236 71 232 91 4
89 767 793 221 307 235 570 762 660 26 548 858 844 867 931 553 360 920 613 878 507 21 141 849 995 283 736 675 462 20 123
645 919 924 178 375 879 215 654 126 331 403 519 260 49 40
```

=====

tempo de execucao: 3.000000

=====

Busca em largura de N = 100

```
Sequencia de visitas em largura:  
100 54 49 68 84 69 3 94 65 28 56 42 85 14 51 29 74 63 37 82 75 96 25 30 24 12 55 15 71 78 87 58 92 62 35 21 19 43 38 73  
70 91 79 27 4 7 88 1 8 5 39 83 41 60 13 59 48 31 2 89 22 72 93 57 11 98 10 40 45 67 97 46 17 18 77 6 36 47 95 23 34 16  
=====  
tempo de execucao: 0.000000  
=====  
Process finished with exit code 0
```

Busca em largura de N = 10

```
Sequencia de visitas em largura:  
10 3 2 1 9 8 7 6 4 5  
=====  
tempo de execucao: 0.000000  
=====  
Process finished with exit code 0  
|
```

Busca em largura de N = 1

```
Sequencia de visitas em largura:  
2 1  
=====  
tempo de execucao: 0.000000  
=====  
Process finished with exit code 0
```

Referências

- [05-ArvoresAVL.pdf \(uff.br\)](#)
- [Árvore AVL – Wikipédia, a enciclopédia livre \(wikipedia.org\)](#)
- [Busca em largura \(BFS\) \(usp.br\)](#)
- [Busca em largura \(pantuza.com\)](#)
- [Busca em largura – Wikipédia, a enciclopédia livre \(wikipedia.org\)](#)