Sorting – Radix sorting – Example

Suppose we want to sort the following sequence with Radix sort

15 43 5 27 60 18 26 2

Binary encodings are given by

15 = 001111	43 = 101011	5 = 000101	27 = 011011
60 = 111100	18 = 010010	26 = 011010	2 = 000010

- items have bit positions 0,…,5, hence m=6
- b must be a factor of m, so lets choose b=2

This means in Radix sort we have:

- $2^b=2^2=4$ buckets labelled 0,1,2,3 (or equivalently 00,01,10,11) and m/b = 3 iterations are required

Sorting – Radix sorting – Example

Sequence:

15 43 5 27 60 18 26 2

Binary encodings:

15 = 001111	43 = 101011	5 = 000101	27 = 011011
60 = 111100	18 = 010010	26 = 011010	2 = 000010

An iteration of radix (after substituting **b** with **2**)

- items are distributed into 4 buckets (a bucket is just a list)
- during the i^{th} iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions $2\times i-1,\ldots,2\times(i-1)$
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

Sorting – Radix sorting – Example

Sequence:

15	43	5	27	60	18	26	2

Binary encodings:

15 = 001111	43 = 101011	5 = 000101	27 = 011011
60 = 111100	18 = 010010	26 = 011010	2 = 000010

First iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 1^{st} iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions $2\times1-1,\ldots,2\times(1-1)$
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

Sorting – Radix sorting – Example

Sequence:

15	43	5	27	60	18	26	2

Binary encodings:

15 = 001111	43 = 101011	5 = 000101	27 = 011011
60 = 111100	18 = 010010	26 = 011010	2 = 000010

First iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 1st iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 1,…,0
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

Sorting – Radix sorting – Example

Sequence:

15 43 5 27 60 18 26 2

Binary encodings:

```
15 = 001111   43 = 101011    5 = 000101   27 = 011011
60 = 111100   18 = 010010   26 = 011010    2 = 000010
```

First iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 1st iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 1,…,0
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
1st iteration:
bucket 00:
bucket 01:
bucket 10:
bucket 11:
new sequence:
```

Sorting – Radix sorting – Example

Sequence:

```
15    43    5    27    60    18    26    2
```

Binary encodings:

```
15 = 001111   43 = 101011    5 = 000101   27 = 011011
60 = 111100   18 = 010010   26 = 011010    2 = 000010
```

First iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 1st iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 1,…,0
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
1st iteration:
bucket 00: 60
bucket 01:  5
bucket 10: 18 26  2
bucket 11: 15 43 27
new sequence:
```

Sorting – Radix sorting – Example

Sequence:

```
15    43    5    27    60    18    26    2
```

Binary encodings:

15 = 001111	43 = 101011	5 = 000101	27 = 011011
60 = 111100	18 = 010010	26 = 011010	2 = 000010

First iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 1st iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 1,…,0
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
1st iteration:
bucket 00: 60
bucket 01:  5
bucket 10: 18 26  2
bucket 11: 15 43 27
new sequence:
```

Sorting – Radix sorting – Example

Sequence:

15 43 5 27 60 18 26 2

Binary encodings:

```
15 = 001111   43 = 101011    5 = 000101   27 = 011011
60 = 111100   18 = 010010   26 = 011010    2 = 000010
```

First iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 1st iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 1,…,0
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
1st iteration:
bucket 00: 60
bucket 01:  5
bucket 10: 18 26  2
bucket 11: 15 43 27
new sequence:
```

Sorting – Radix sorting – Example

Sequence:

15 43 5 27 60 **18** **26** **2**

Binary encodings:

```
15 = 001111   43 = 101011    5 = 000101   27 = 011011
60 = 111100   18 = 010010   26 = 011010    2 = 000010
```

First iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 1st iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 1,…,0
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
1st iteration:
bucket 00: 60
bucket 01:  5
bucket 10: 18 26  2
bucket 11: 15 43 27
new sequence:
```

Sorting – Radix sorting – Example

Sequence:

```
15   43   5   27   60   18   26   2
```

Binary encodings:

```
15 = 001111   43 = 101011    5 = 000101   27 = 011011
60 = 111100   18 = 010010   26 = 011010    2 = 000010
```

First iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 1st iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 1,…,0
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
1st iteration:
bucket 00: 60
bucket 01:  5
bucket 10: 18 26  2
bucket 11: 15 43 27
new sequence:
```

Sorting – Radix sorting – Example

Sequence:

```
15    43    5    27    60    18    26    2
```

Binary encodings:

```
15 = 001111   43 = 101011    5 = 000101   27 = 011011
60 = 111100   18 = 010010   26 = 011010    2 = 000010
```

First iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 1st iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 1,…,0
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
1st iteration:
bucket 00: 60
bucket 01:  5
bucket 10: 18 26  2
bucket 11: 15 43 27
new sequence: 60 5 18 26 2 15 43 27
```

Sorting – Radix sorting – Example

New sequence: 60 5 18 26 2 15 43 27

Binary encodings:

60 = 111100	5 = 000101	18 = 010010	26 = 011010
2 = 000010	15 = 001111	43 = 101011	27 = 011011

Second iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 2nd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions $2\times2-1,\dots,2\times(2-1)$
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

Sorting – Radix sorting – Example

New sequence: 60 5 18 26 2 15 43 27

Binary encodings:

60 = 111100	5 = 000101	18 = 010010	26 = 011010
2 = 000010	15 = 001111	43 = 101011	27 = 011011

Second iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 2nd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions $3,\ldots,2$
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

Sorting – Radix sorting – Example

New sequence:

```
60     5     18     26     2     15     43     27
```

Binary encodings:

```
60 = 111100    5 = 000101   18 = 010010    26 = 011010
 2 = 000010   15 = 001111   43 = 101011    27 = 011011
```

Second iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 2nd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 3,…,2
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
2nd iteration:
bucket 00:
bucket 01:
bucket 10:
bucket 11:
new sequence:
```

Sorting – Radix sorting – Example

New sequence:

```
60     5     18     26     2     15     43     27
```

Binary encodings:

```
60 = 111100    5 = 000101   18 = 010010    26 = 011010
 2 = 000010   15 = 001111   43 = 101011    27 = 011011
```

Second iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 2nd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 3,…,2
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
2nd iteration:
bucket 00: 18  2
bucket 01:  5
bucket 10: 26 43 27
bucket 11: 60 15
new sequence:
```

Sorting – Radix sorting – Example

New sequence:

```
60    5    18    26    2    15    43    27
```

Binary encodings:

```
60 = 111100    5 = 000101   18 = 010010    26 = 011010
 2 = 000010   15 = 001111   43 = 101011    27 = 011011
```

Second iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 2nd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 3,…,2
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
2nd iteration:
bucket 00: 18  2
bucket 01:  5
bucket 10: 26 43 27
bucket 11: 60 15
new sequence:
```

Sorting – Radix sorting – Example

New sequence:

```
60   5   18   26   2   15   43   27
```

Binary encodings:

```
60 = 111100    5 = 000101   18 = 010010   26 = 011010
 2 = 000010   15 = 001111   43 = 101011   27 = 011011
```

Second iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 2nd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 3,…,2
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
2nd iteration:
bucket 00: 18  2
bucket 01:  5
bucket 10: 26 43 27
bucket 11: 60 15
new sequence:
```

Sorting – Radix sorting – Example

New sequence:

```
60    5    18    26    2    15    43    27
```

Binary encodings:

```
60 = 111100    5 = 000101   18 = 010010   26 = 011010
 2 = 000010   15 = 001111   43 = 101011   27 = 011011
```

Second iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 2^{nd} iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions $3,\ldots,2$
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
2nd iteration:
bucket 00: 18  2
bucket 01:  5
bucket 10: 26 43 27
bucket 11: 60 15
new sequence:
```

Sorting – Radix sorting – Example

New sequence:

60	5	18	26	2	**15**	43	27

Binary encodings:

60 = 111100	5 = 000101	18 = 010010	26 = 011010
2 = 000010	**15 = 001111**	43 = 101011	27 = 011011

Second iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 2nd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 3,…,2
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
2nd iteration:
bucket 00: 18  2
bucket 01:  5
bucket 10: 26 43 27
bucket 11: 60 15
new sequence:
```

Sorting – Radix sorting – Example

New sequence:

```
60    5    18    26    2    15    43    27
```

Binary encodings:

```
60 = 111100    5 = 000101   18 = 010010    26 = 011010
 2 = 000010   15 = 001111   43 = 101011    27 = 011011
```

Second iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 2nd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 3,…,2
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
2nd iteration:
bucket 00: 18  2
bucket 01:  5
bucket 10: 26 43 27
bucket 11: 60 15
new sequence: 18 2 5 26 43 27 60 15
```

Sorting – Radix sorting – Example

New sequence:

18	2	5	26	43	27	60	15

Binary encodings:

18	=	010010	2	=	000010	5	=	000101	26	=	011010
43	=	101011	27	=	011011	60	=	111100	15	=	001111

Third (and final) iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 3rd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions $2\times3-1,\ldots,2\times(3-1)$
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

Sorting – Radix sorting – Example

New sequence: 18 2 5 26 43 27 60 15

Binary encodings:

18 = 010010	2 = 000010	5 = 000101	26 = 011010
43 = 101011	27 = 011011	60 = 111100	15 = 001111

Third (and final) iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 3rd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 5,…,4
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

Sorting – Radix sorting – Example

New sequence:

```
18   2   5   26   43   27   60   15
```

Binary encodings:

```
18 = 010010    2 = 000010    5 = 000101   26 = 011010
43 = 101011   27 = 011011   60 = 111100   15 = 001111
```

Third (and final) iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 3rd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 5,…,4
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
3rd iteration:
bucket 00:
bucket 01:
bucket 10:
bucket 11:
sorted sequence:
```

Sorting – Radix sorting – Example

New sequence:

```
18    2    5    26    43    27    60    15
```

Binary encodings:

```
18 = 010010    2 = 000010    5 = 000101   26 = 011010
43 = 101011   27 = 011011   60 = 111100   15 = 001111
```

Third (and final) iteration of radix

– items are distributed into 4 buckets (a bucket is just a list)

– during the 3rd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 5,…,4

– buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
3rd iteration:
bucket 00:  2  5 15
bucket 01: 18 26 27
bucket 10: 43
bucket 11: 60
sorted sequence:
```

Sorting – Radix sorting – Example

New sequence: 18 **2** **5** 26 43 27 60 **15**

Binary encodings:

18 = 010010	**2 = 000010**	**5 = 000101**	26 = 011010
43 = 101011	27 = 011011	60 = 111100	**15 = 001111**

Third (and final) iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 3rd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 5,…,4
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
3rd iteration:
bucket 00:  2  5 15
bucket 01: 18 26 27
bucket 10: 43
bucket 11: 60
sorted sequence:
```

Sorting – Radix sorting – Example

New sequence **18** 2 5 **26** 43 **27** 60 15

Binary encodings:

18 = 010010 2 = 000010 5 = 000101 **26 = 011010**
43 = 101011 **27 = 011011** 60 = 111100 15 = 001111

Third (and final) iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 3rd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 5,…,4
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
3rd iteration:
bucket 00:  2  5 15
bucket 01: 18 26 27
bucket 10: 43
bucket 11: 60
sorted sequence:
```

Sorting – Radix sorting – Example

New sequence:

```
18    2    5    26    43    27    60    15
```

Binary encodings:

```
18 = 010010   2 = 000010   5 = 000101   26 = 011010
43 = 101011  27 = 011011  60 = 111100   15 = 001111
```

Third (and final) iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 3rd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 5,…,4
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
3rd iteration:
bucket 00:  2  5 15
bucket 01: 18 26 27
bucket 10: 43
bucket 11: 60
sorted sequence:
```

Sorting – Radix sorting – Example

New sequence:

```
18    2    5    26    43    27    60    15
```

Binary encodings:

```
18 = 010010    2 = 000010    5 = 000101   26 = 011010
43 = 101011   27 = 011011   60 = 111100   15 = 001111
```

Third (and final) iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 3rd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 5,…,4
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
3rd iteration:
bucket 00:  2  5 15
bucket 01: 18 26 27
bucket 10: 43
bucket 11: 60
sorted sequence:
```

Sorting – Radix sorting – Example

New sequence:

```
18    2    5    26    43    27    60    15
```

Binary encodings:

```
18 = 010010    2 = 000010    5 = 000101    26 = 011010
43 = 101011   27 = 011011   60 = 111100    15 = 001111
```

Third (and final) iteration of radix

- items are distributed into 4 buckets (a bucket is just a list)
- during the 3rd iteration, an item is placed in a bucket corresponding to the integer represented by the bits in positions 5,…,4
- buckets concatenated at the end of an iteration to give input sequence for the next iteration

```
3rd iteration:
bucket 00:  2  5 15
bucket 01: 18 26 27
bucket 10: 43
bucket 11: 60
sorted sequence: 2 5 15 18 26 27 43 60
```